LEAF "Bering" user's guide

LEAF "Bering" user's guide

Bering users Community

Edited by J. Nilo & E. Wolzak

Revision History

Revision 0.1	15 March 2002
First draft for review	
Revision 0.2	14 April 2002
Second draft for review	
Revision 0.3	18 May 2002
Third draft for review	
Revision 0.4	16 June 2002
Fourth draft for review	
Revision 0.5	20 October 2002
Fith draft for review	

Table of Contents

1. Structure of the document
- *1.1. Overview*
- *1.2. Contributions and Feedback*
- *1.3. Changelog*

2. Serial Modem configuration
- *2.1. Objectives*
- *2.2. Step 1: declare the ppp package*
- *2.3. Step 2: declare the ppp modules*
- *2.4. Step 3: configure ppp*
- *2.5. Step 4: configure your interfaces file*
- *2.6. Step 5: configure Shorewall*
- *2.7. Step 6: reboot...*

3. PCMCIA configuration
- *3.1. Objectives*
- *3.2. Step 1: declare the ppp and the pcmcia packages*
- *3.3. Step 2: declare the ppp modules in modules.lrp*
- *3.4. Step 3: configure ppp*
- *3.5. Step 4: configure pcmcia*
- *3.6. Step 5: configure your interfaces file*
- *3.7. Step 6: configure Shorewall*
- *3.8. Step 7: reboot...*

4. PPPoE configuration
- *4.1. Objectives*
- *4.2. Step 1: declare the ppp and pppoe packages*
- *4.3. Step 2: declare the ppp and pppoe modules*
- *4.4. Step 3: configure ppp*

4.5. Step 4: Configure pppoe
4.6. Step 5: configure your interfaces file
4.7. Step 6: configure Shorewall
4.8. Step 7: reboot...
4.9. An example: a PPPoE connection with a two PCMCIA cards setup
5. PPPoA configuration
5.1. Objectives
5.2. Step 1: declare the pppatm package
5.3. Step 2: declare the ppp and pppoatm modules
5.4. Step 3: configure pppatm
5.5. Step 4: configure your interfaces file
5.6. Step 5: configure Shorewall
5.7. Step 7: reboot...
6. PPTP/PPPoA configuration
6.1. Objectives
6.2. Step 1: declare the ppp and the pptp packages
6.3. Step 2: declare the ppp modules
6.4. Step 3: configure ppp
6.5. Step 4: configure your interfaces file
6.6. Step 5: configure Shorewall
6.7. Step 7: reboot...
7. ISDN Configuration
7.1. Objectives
7.2. Step 1: Download and declare the isdn.lrp package
7.3. Step 2: download the isdn.o and the appropriate hisax.o modules
7.4. Step 3: declare the ISDN modules
7.5. Step 4: configure ISDN
7.6. Step 5: configure your interfaces file
7.7. Step 6: configure Shorewall
8. Creating a bootable Bering CD−ROM
8.1. Objectives
8.2. Step 1: preparing the distro
8.3. Step 2: downloading the required packages
8.4. Step 3: preparing the Bering CD−Rom content
8.5. Step 4: making the CD
8.6. Support
8.7. Thanks to...
9. Booting Bering from different boot−media
9.1. Objectives
9.2. The single floppy drive setup
9.3. The two−floppy drives setup
9.4. Booting from an IDE device
9.5. Booting from a CD−Rom with isolinux
9.6. Partial backup of packages to/from floppy
10. Installing and booting Bering from a M−Systems DiskOnChip
10.1. Objectives
10.2. Step 1: prepare the boot floppy
10.3. Step 2: apply bug fixes
10.4. Step 3: configure Bering for DoC booting
10.5. Step 4: prepare the DoC
10.6. Step 5: reboot

10.7. Thanks to...
11. Wireless and orinoco drivers
11.1. Objectives
11.2. Step 1: declare the appropriate packages
11.3. Step 2: declare the appropriate modules
11.4. Step 3: configure ppp
11.5. Step 4: configure pcmcia and wireless
11.6. Step 5: configure your interfaces file
11.7. Step 6: configure Shorewall
11.8. Tips and tricks
12. IPSec configuration
12.1. Objectives
12.2. Step 1: load ipsec or ipsec509 package
12.3. Step 2: generate certificates with openssl
12.4. Step 3: boot Bering and move certificates into place
12.5. Step 4: configure ipsec.conf
12.6. Step 5: configure ipsec.secrets
12.7. Step 6: configure Shorewall
12.8. Step 7: configure Windows 2000 client
13. Monitoring Bering through a terminal console
13.1. Objectives
13.2. Step 1: Modify `/etc/inittab` and `/etc/securetty` files
13.3. Step 2: Modify your `syslinux.cfg` file
13.4. Step 3: reboot...
14. Time in Bering
14.1. Objectives
14.2. Define your timezone
14.3. Set the system date/time
14.4. Edit the contents of `/etc/timezone` (optional)
14.5. Activate daily clock updating (optional)
14.6. Internal network NTP clients
14.7. Miscellaneous
15. The Bering "mail" and "cron" facilities
15.1. Objectives
15.2. The mail command
15.3. Cronjobs

Prev Home Next
Structure of the document

1. Structure of the document

1.1. Overview

The LEAF "Bering" user's guide is organized around practical problems (and hopefully solutions) encountered by many Bering users. Users contributions are encouraged and welcomed. They can be send to the authors either in plain ASCII form or − better − in Docbook XML format. XML source code are available to everyone and can be used as templates.

Basic prior knowledge of linux and of the LEAF Bering distro (or any other LEAF distributions like Dachstein or Oxygen) is assumed. In particular the reader is supposed to be able to perform the following tasks:

- Add or remove a package to/from a LEAF distribution through editing of the floppy `syslinux.cfg` file and move it to (out of) the Bering floppy disk
- Add or remove a Bering linux kernel module by moving it to (out of) `/lib/modules` or `/boot/lib/modules` directory
- Adjust the parameters of a given package through the LEAF configuration menu and backup a package

The following reference is a prerequisite reading:

- The Bering Installation guide

1.2. Contributions and Feedback

Contributions to and comments on this document can be sent to the authors:

Jacques Nilo <jnilo@users.sourceforge.net> or Eric Wolzak <leaf@wolzak.de>.

You can download the docbook xml sources from the different sections of this user's guide here to be used as a template. A complete Docbook XML documentation can be found here.

1.3. Changelog

Current version: 0.5 − October 2002

Added following sections:

- Installing and booting Bering from a M−Systems DiskOnChip (B. Fritz)
- Time in Bering (J. Nilo & E. Wolzak)
- The Bering "mail" and "cron" facilities (E. Wolzak)

Version: 0.4 − June 2002

Various sections edited for typos and updates

Version: 0.3 − May 2002

Added following section:

- PPPoA configuration (J. Nilo)

Version: 0.2 − April 2002

Added following sections:

- Creating a bootable Bering CD−Rom (L. Correia)
- Booting Bering from different boot medias (J.Nilo/E. Wolzak)
- Wireless and orinoco drivers (J. Nilo)
- IPSEC configuration (C. Carr)
- PPTP/PPPoA configuration (J. Nilo)
- Monitoring Bering through a terminal console (J. Nilo)

Serial Modem, PCMCIA, PPPoE and ISDN sections corrected and edited.

Version: 0.1 − March 2002

Added following sections:

- Serial Modem configuration (J. Nilo)
- PCMCIA configuration (J. Nilo)
- PPPoE configuration (E. Wolzak)
- ISDN configuration (E. Wolzak)

Prev | Home | Next
LEAF "Bering" user's guide | Up | Serial Modem configuration

2. Serial Modem configuration

2.1. Objectives

We assume here that you can only get connected to internet through a serial modem connection and that you want to share that connection with other (internal) computers in your home or office. What follows describe the configuration of this dial−up modem router. Your external interface (to the internet) will be ppp0, your internal interface (to your internal network) is supposed to be done through an ethernet network card (eth0).

What follows has been tested with Bering v1.0−rc1 on a Pentium 133 machine and a US Robotics external modem connected to com1 (ttyS0).

The PPP−Howto is a useful reference for this section.

Comments on this section should be addressed to its maintainer: Jacques Nilo <jnilo@users.sourceforge.net>. Thanks to Lee who provided useful additions to this section.

2.2. Step 1: declare the ppp package

Boot a Bering floppy image. Once the LEAF menu appears get access to the linux shell by (q)uitting the menu. Edit the `syslinux.cfg` file and replace the pump entry by ppp in the LRP= list of packages to be loaded at boot. Check the Bering installation guide to learn how to do that.

Your `syslinux.cfg` file could look like (adjust to your tastes):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,ppp,keyboard,shorwall,dnscache,weblet
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

2.3. Step 2: declare the ppp modules

In order to have a modem dialup connection working, you need to have ppp support enabled through the appropriate kernel modules (note: since v1.0−rc2, serial support is compiled in the kernel). You also need to declare the driver module of the network card assigned to your internal network. In the following example, this card is supposed to be a standard ne 2000 PCI card.

To configure your modules, go to the LEAF Packages configuration menu and choose modules. Enter 1) to edit the `/etc/modules` file and enter the following information:

```
# 8390 based ethernet cards
8390
ne2k−pci

# Modules needed for PPP connection
```

```
slhc
ppp_generic
ppp_async
ppp_deflate

# Masquerading 'helper' modules
ip_conntrack_ftp
ip_conntrack_irc
ip_nat_ftp
ip_nat_irc
```

The sample file above might be different in your own case: you might need another network module or some extra functionnalities. Adjust to your needs !

Backup the modules.lrp package.

2.4. Step 3: configure ppp

Connection with your ISP will be handled by PPP. The PPP How−to document will give you very detailed information about this protocol and how to set−up the numerous parameters.

Through the LEAF packages configuration menu get access to ppp configuration. The following menu will show−up

```
                        ppp configuration files

       1) ISP pppd options
       2) ISP login script
       3) System wide pppd options
       4) chap secret
       5) pap secret
       6) pppd daemon script

  q) quit
  ------------------------------------------------------------------------------
        Selection:
```

Entry 1) allows you to adjust the parameter of your ppp connection through the `/etc/ppp/peers/provider` file. The most important argument is the *ttySx* parameter which defines the serial port to which your modem is connected.

Look at your `/var/log/syslog` file after booting Bering. It will give you the list of the serial ports recognized by your linux kernel.

A working `/etc/ppp/peers/provider` file for a Compuserve connection could look like:

```
# ISP pppd options file
# What follows is OK for Compuserve
#
noauth
debug           # log transaction to /var/log/messages
/dev/ttyS0      # (ttyS0=com1, ttyS1=com2, ...)
115200          # baud  rate
modem
```

```
crtscts         # use hardware flow control
asyncmap 0
defaultroute    # ppp becomes default route to the internet
noipdefault
lock            # don't let other processes besides PPP use the device
connect "/usr/sbin/chat −v −f /etc/chatscripts/provider"
```

If you plan to dial into a Windows RAS server or a server that uses PAP or CHAP authentication, you need to add a line to this file. Just above the "connect" command, on a line of its own, add "name <ISPUserID>" where <ISPUserID> is the login name your ISP gave you. You need this because ppp has to masquerade the firewall as you when using PAP or CHAP authentication.

Entry 2) allows you to adjust the communication script which will handle the connection with your ISP. This script is stored in the `/etc/chatscripts/provider`

If you are not using Compuserve you should also delete all of the lines below the "comment" line. A few − very few − ISPs require the final "PPP" line these days.

A working script for a Compuserve connection could look like:

```
# ISP login script
# What follows is OK for Compuserve
# Adjust to your taste
ABORT "BUSY"
ABORT "NO CARRIER"
ABORT "VOICE"
ABORT "NO DIALTONE"
ABORT "NO ANSWER"
"" ATZ
# ISP telephone number: 124567890
OK ATDT1234567890#
CONNECT ''
Name: CIS
# With compuserve your_login_account=12345,6789
ID: your_login_account/go:pppconnect
Password: your_password
PPP
```

Edit Entry 3) − /etc/ppp/options "System−wide pppd options" if you want the system to demand dial and to drop the line if idle for a preset time. To do this, change "persist" to "demand" and add another line below "demand" that says "idle 600", where 600 is the number of seconds the system should wait before dropping hanging up if there is no network traffic.

Edit either the PAP (Entry 4) or CHAP (Entry 5) option to set up how your system authenticates. For PAP authentication, choose the PAP option and add a line saying "<ISPUserID> * <ISPUserPassword> to the bottom of the file. <ISPUserID> is the same entry that you made in Entry 1) − the "ISP pppd ptions" file. The <ISPUserPassword> entry is self−explanatory. The "*" can be replaced with the IP address or name of the server you are dialling into if you know it. Usually, an asterisk is sufficient. If you want to authenticate using CHAP, add the same entry to the CHAP item instead.

Backup the ppp.lrp package.

2.5. Step 4: configure your interfaces file

Trough the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo ppp0 eth0

iface lo inet loopback

iface ppp0 inet ppp
        provider provider

iface eth0 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
```

The "auto" statement declares all the interfaces that will be automatically set up at boot time. This job will be carried out by the "ifup −a" statement in the `/etc/init.d/networking` script.

The syntax of "iface" statements is explained in the Bering's installation guide.

Backup the etc.lrp package.

2.6. Step 5: configure Shorewall

Through the LEAF packages configuration menu, choose shorwall and check the two following files:

A/ The `interfaces` file (entry 3) defines your interfaces. Here connection to the net goes through ppp0 and the connection to the internal network through eth0. So we must set:

```
(...)
#ZONE   INTERFACE       BROADCAST       OPTIONS
net     ppp0            -
loc     eth0            detect          routestopped
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Do not forget the "−" under the BROADCAST heading for the net/ppp0 entry.

B/ The `masq` file (entry 7). With a dial−up modem setup it should look like:

```
(...)
#INTERFACE              SUBNET
ppp0                    eth0
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Backup the shorwall.lrp package.

2.7. Step 6: reboot...

Your modem connection should be established automatically. Type *plog* to check the login sequence with your ISP. If there is no output check `/var/log/syslog` to get a clue on potential problems.

If you want to be sure that your modem and/or script parameters are OK before backing up ppp.lrp, you can launch the connection manually just by typing *pon*. Use the *plog* command to see how the connection is going and *poff* to close down your ppp connection.

Prev | Home | Next
Structure of the document | Up | PCMCIA configuration

3. PCMCIA configuration

3.1. Objectives

We assume here that your cable/ADSL connection is down and that you need to setup a router on your old laptop equipped with a combo Ethernet/Modem PCMCIA card. What follows describe the configuration of this emergency dial−up modem router. Your external interface (to the internet) will be using the modem facility of your PCMCIA card whereas your internal interface (to your internal network) will be connected to the ethernet network plug of your PCMCIA card (eth0).

What follows has been tested with Bering v1.0−rc1 and the pmcia_xircom.lrp package on a NEC Versa SX using a Xircom RealPort EThernet 10/100 + Modem 56k − ref REM56G−100BTX.

The PCMCIA−Howto and the PPP−Howto are useful references for this section.

Comments on this section should be addressed to its maintainer: Jacques Nilo <jnilo@users.sourceforge.net>.

3.2. Step 1: declare the ppp and the pcmcia packages

Boot a Bering floppy image. Once the LEAF menu appears get access to the linux shell by (q)uitting the menu. Edit the `syslinux.cfg` file and replace the pump entry by ppp,pcmcia in the LRP= list of packages to be loaded at boot. Check the Bering installation guide to learn how to do that.

Your `syslinux.cfg` file could look like (adjust to your taste):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,ppp,pcmcia,keyboard,shorwall,dnscache,
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

The ppp package is provided on the standard Bering floppy. The pcmcia.lrp package is available in the Bering download packages area. Check the Bering installation guide.

3.3. Step 2: declare the ppp modules in modules.lrp

In order to have a modem dialup connection working, you need to have ppp support enabled through the appropriate kernel modules.

To configure your modules, go to the LEAF Packages configuration menu and choose modules. Enter 1) to edit the `/etc/modules` file and enter the following information:

```
# Modules needed for PPP connection
slhc
```

```
ppp_generic
ppp_async
ppp_deflate

# Masquerading 'helper' modules
ip_conntrack_ftp
ip_conntrack_irc
ip_nat_ftp
ip_nat_irc
```

Backup the modules.lrp package.

3.4. Step 3: configure ppp

Connection with your ISP will be handled by PPP. The PPP How−to document will give you very detailed information about this protocol and how to set−up its numerous parameters.

Please refer to the Serial Modem section of this user's guide to learn how to configure your ppp package.

3.5. Step 4: configure pcmcia

First make sure to install in your pcmcia package the PCMCIA kernel modules that will be needed by your hardware. Refer to the Bering installation guide to learn how to do that.

For our Xircom card the following modules were used:

```
# ls −la /lib/modules/pcmcia
drwxr−xr−x    2 root     root         4096 avr 25 07:54 ./
drwxrwxrwt   27 root     root         4096 avr 25 07:52 ../
−rw−r−−r−−    1 root     root        11248 avr 25 07:53 ds.o
−rw−r−−r−−    1 root     root        33728 avr 25 07:53 i82365.o
−rw−r−−r−−    1 root     root        57272 avr 25 07:54 pcmcia_core.o
−rw−r−−r−−    1 root     root         8204 avr 25 07:54 serial_cs.o
−rw−r−−r−−    1 root     root        19680 avr 25 07:54 xirc2ps_cs.o
```

Once your package is ready, enter the LEAF Package configuration menu and choose pcmcia. The following menu will appear

```
                      pcmcia configuration files

       1) pcmcia default parameters
       2) pcmcia configuration
       3) wireless configuration

  q) quit
  ------------------------------------------------------------------------
       Selection:
```

Entry 1) allows to edit the `/etc/default/pcmcia` file which defines the pcmcia parameters that will be used by the cardmgr program and the `/etc/init.d/pcmcia` script.

In our practical example (a Xircom RealPort EThernet 10/100 + Modem 56k − ref REM56G−100BTX) this file will contain:

```
PCMCIA=yes
PCIC=i82365.o
PCIC_OPTS=
CORE_OPTS=
CARDMGR_OPTS=
```

Entry 2) allows to edit the `/etc/pcmcia/config.opts` file. Please refer to the PCMCIA How−to for the explanation of the different options.

The default `/etc/pcmcia/config.opts` file provided in the pcmcia.lrp package is the default file provided in the pcmcia−cs package. It looks like:

```
include port 0xa00−0xaff
#
# Resources we should not use, even if they appear to be available
#
# First built−in serial port
exclude irq 4
# Second built−in serial port
exclude irq 12
# First built−in parallel port
exclude irq 7
```

Entry 3) is only used if you are using a wireless PCMCIA card. If not this file can only contain:

```
*,*,*,*)
    ;;
```

Refer to the wireless section of this user's guide if you need to setup wireless.

3.6. Step 5: configure your interfaces file

Trough the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo

iface lo inet loopback

iface eth0 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
up pon
up /etc/init.d/dnscache restart
up shorewall restart
down shorewall stop
down /etc/init.d/dnscache stop
down poff
```

No interface (except lo) is activated automatically. The pcmcia package will start cardmgr through the `/etc/init.d/pcmcia` script executed at boot time. The cardmgr program will then call the `/etc/pcmcia/network` script which will activate the eth0 interface using the information from the `/etc/network/interfaces` file. Here, the `/etc/network/interfaces` says for eth0:

- Assign ip address 192.168.1.254/24 to the interface
- Once eth0 is up, start the ppp connection through the `pon` script
- Then restart dnscache, since dnscache was unable to start at boot time, eth0 being not available at that time
- Then restart shorewall for the same reason

When stopping pcmcia the same command are executed in the reverse order through the down statement.

Backup the etc.lrp package.

3.7. Step 6: configure Shorewall

Through the LEAF packages configuration menu, choose shorwall and check the two following files:

A/ The `interfaces` file (entry 3) defines your interfaces. Here connection to the net goes through ppp0. So we must set:

```
(...)
#ZONE   INTERFACE       BROADCAST       OPTIONS
net     ppp0            -
loc     eth0            detect          routestopped
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Do not forget the "−" under the BROADCAST heading for the net/ppp0 entry.

B/ The `masq` file (entry 7). In this context it should look like:

```
(...)
#INTERFACE              SUBNET
ppp0                    eth0
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Backup the shorwall.lrp package.

3.8. Step 7: reboot...

Your modem connection should be established automatically. Type *plog* to check the login sequence with your ISP. If there is no output check `/var/log/syslog` to get a clue on potential problems.

If you want to be sure that your modem and/or script parameters are OK before backing up ppp.lrp and/or pcmcia.lrp, you can activate the pcmcia connection manually by typing launch the connection manually just by typing `/etc/init.d/pcmcia start`. Use `/etc/init.d/pcmcia stop` to stop the pcmcia connection, remove the modules and bring down eth0 and ppp0.

4. PPPoE configuration

4.1. Objectives

We assume here that you want to connect your LEAF router to the Internet via an ADSL PPPoE connection. What is described here corresponds to section 3.2.3 of the DSL How−To document. Your ADSL modem is supposed to be connected to eth0, while the traffic to your internal network goes through eth1.

What follows has been tested with Bering v1.0−rc1 on a 486er and a pentium machine, a rtl8139 compatible and a 3com network card connected to eth0 and eth1 and ADSL "T−online" service offered here in Germany.

The PPP−Howto and the DSL−Howto are two useful references for this section.

Comments on this section should be addressed to its maintainer: Eric Wolzak <leaf@wolzak.de>.

4.2. Step 1: declare the ppp and pppoe packages

Those two packages are provided on the standard Bering floppy disk, but are not activated by default.

Boot a Bering floppy image. Once the LEAF menu appears get access to the linux shell by (q)uitting the menu. Edit the `syslinux.cfg` file and REPLACE the pump entry by ppp,pppoe in the LRP= list of packages to be loaded at boot. Check the Bering installation guide to learn how to do that.

Your `syslinux.cfg` file will then look like (adjust to your tastes):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,ppp,pppoe,keyboard,shorwall,dnscache,w
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

4.3. Step 2: declare the ppp and pppoe modules

In order to have a PPPoE connection working, you need to have ppp and pppoe support enabled through the appropriate kernel modules. You also need to declare the driver(s) module(s) of your network card(s). In the following example, we assume that both ethernet interfaces are provided through a standard ne 2000 PCI card.

All the modules which are necessary for a PPPoE connection are provided on the standard Bering floppy. You just need to "declare" them since they are not loaded by default. As far as your network cards are concerned, the most popular driver modules are provided in `/lib/modules` but you might need to download the one corresponding to your own hardware from the Bering modules download area. Refer to the Bering installation guide to learn how to do that.

To declare your modules, go to the LEAF Packages configuration menu and choose modules. Enter 1) to edit the `/etc/modules` file and enter the following information:

```
# 8390 based ethernet cards
8390
ne2k-pci

# Modules needed for PPP/PPPOE connection
slhc
n_hdlc
ppp_generic
ppp_synctty
pppox
pppoe

# Masquerading 'helper' modules
ip_conntrack_ftp
ip_conntrack_irc
ip_nat_ftp
ip_nat_irc
```

The `/etc/modules` file provided in the Bering distro is already setup with those entries commented out. Just remove the leading # sign to activate the corresponding module.

Backup the modules.lrp package.

4.4. Step 3: configure ppp

Connection with your ISP will be handled by PPP. The PPP Howto document will give you very detailed information about this protocol and how to set-up its numerous parameters.

Please refer to the Serial Modem configuration section of this user's guide to learn how to configure your ppp package.

The default options provided with the ppp.lrp should work and if you are not familiar with ppp leave them at first. After you get a connection you can "fine tune" your setup.

4.5. Step 4: Configure pppoe

Through the LEAF Package configuration menu choose pppoe. The following menu will appear:

```
                      pppoe configuration files

      1) DSL pppd options
      2) pap secret

 q) quit
 --------------------------------------------------------------------------
      Selection:
```

Entry 1) allows you to adjust the parameter of your ppp connection through the `/etc/ppp/peers/dsl-provider` file. The most important argument is the *name* parameter which defines your login name.

Replace the field following the name statement in the `/etc/ppp/peers/dsl-provider` ["login@isp.com"] by the login name provided by your ISP.

```
# Configuration file for PPP, using PPP over Ethernet
# to connect to a DSL provider.
#
plugin /usr/lib/pppd/pppoe.so

# MUST CHANGE: Uncomment the following line, replacing the user@provider.net
# by the DSL user name given to your by your DSL provider.
# (There should be a matching entry in /etc/ppp/pap-secrets with the password.)
user "eric12345@foobar.com"

(...)
```

Entry 2) allows you to edit the `/etc/ppp/pap-secrets`. Enter in this file the login and password provided by your ISP. Your login name must EXACTLY match the one given in the previous `/etc/ppp/peers/dsl-provider` file. If you have special characters in secret or username, you should put them in quotes

```
# This is a pap-secrets file
#
#papname * papsecret
"eric12345@foobar.com" * "secretfoo"
```

Backup both pppoe and ppp packages.

4.6. Step 5: configure your interfaces file

Trough the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo ppp0 eth1

iface lo inet loopback

iface ppp0 inet ppp
        pre-up ip link set eth0 up
        provider dsl-provider eth0

iface eth1 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
```

In this `/etc/network/interfaces` file the lo, ppp0 and eth1 interfaces are brought up automatically when the *ifup -a* statement is executed at boot time by the `/etc/init.d/networking` script.

The "iface ppp0 inet ppp" says:

- Execute the "ip link set eth0 up" command BEFORE ppp0 is activated (pre-up statement)
- Execute the `/sbin/pon dsl-provider eth0` script to establish the PPPoE connection. The `dsl-provider` file used as input by `/sbin/pon` is provided in the pppoe.lrp package.

The "iface eth1 inet static" defines the internal address of the router.

Backup the etc.lrp package.

4.7. Step 6: configure Shorewall

Through the LEAF packages configuration menu, choose shorwall and check the three following files:

A/ The `interfaces` file (entry 3) defines your interfaces. Here connection to the net goes through ppp0. So we must set:

```
(...)
#ZONE   INTERFACE       BROADCAST       OPTIONS
net     ppp0            -               routefilter
loc     eth1            detect          routestopped
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Do not forget the "−" under the BROADCAST heading for the net/ppp0 entry.

B/ The `masq` file (entry 7). With a dial−up modem setup it should look like:

```
(...)
#INTERFACE              SUBNET
ppp0                    eth1
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

C/ You may also need to edit the `config` file (entry 12) to adjust the CLAMPMSS variable to "yes":

```
(...)
# Set this variable to "Yes" or "yes" if you want the TCP "Clamp MSS to PMTU"
# option. This option is most commonly required when your internet
# interface is some variant of PPP (PPTP or PPPoE). Your kernel must
#
# If left blank, or set to "No" or "no", the option is not enabled.
#
CLAMPMSS="yes"
(...)
```

Backup the shorwall.lrp package.

4.8. Step 7: reboot...

Your modem connection should be established automatically. Type *plog* to check the login sequence with your ISP. If there is no output check `/var/log/syslog` to get a clue on potential problems.

PPPoE connections are going up and taken down. Here my provider takes down the connection after 15 minutes of inactivity. Also if you switch your router out over night and wants to know if it really got connected beep.lrp is your friend. It gives a sound of configurable duration and frequency. If you have your router on a greater distance, have a monitor installed or use the serial line for direct monitoring, you don't use it. The package should only be inserted on the disk and beep written in the syslinux.cfg package=.....,beep. The configuration is easy, in /etc/ppp/if−up there is allready a small sound included. You can change frequency with the −f option.

4.9. An example: a PPPoE connection with a two PCMCIA cards setup

C. Hostelet is using an old laptop as a Bering router. His hardware configuration consists of one HP Omnibook 3000 laptop (Pentium 233Mhz, 144MB Ram, CD−Rom drive module, no floppy, no HDD), one Xircom CEM56 Modem/ethernet PCMCIA card and one 3Com 3C589 PCMCIA card. The connection to the net is provided through the first PCMCIA card connected to an Alcatel SpeedTouch Home ethernet modem which gives him access to France Telecom "Netissimo" ADSL service. The connection to the local network is done trough the second PCMCIA card.

Here is his `/etc/network/interfaces` file:

```
auto lo

iface lo inet loopback

iface eth0 inet static
        address 10.0.0.1
        masklen 24
        broadcast 10.0.0.255
        up pon dsl-provider eth0
        up shorewall restart
        down shorewall stop
        down poff

iface eth1 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
        up /etc/init.d/dnscache restart
        down /etc/init.d/dnscache stop
```

Only lo is brought up automatically at boot time. eth0 and eth1 are brought up by the PCMCIA cardmgr program which calls the `/etc/pcmcia/network` script.

The connection with the Alcatel speedtouch modem is done through the eth0 interface at address 10.0.0.1

Once the eth0 interface is up the pppd daemon is called by the pon script. Shorewall must then be restarted since eth0 was not available at boot time

Once the eth1 interface is up we restart dnscache which could not start at boot time since eth1 was not available.

5. PPPoA configuration

5.1. Objectives

We assume here that you want to connect your LEAF router to the Internet via PPPoA. The PPPoE connection is covered in another section of this user's guide. So is the PPTP/PPPoA connection. What is described here corresponds to section 3.2.4 of the DSL How−To document. The traffic to your internal network goes through eth0 while access to the Internet via PPPoA goes through ppp0.

The PPP−Howto and the DSL−Howto are two useful references for this section.

The following setup has been tested by Dave Anderson, who gets connected to BT DSL service using a Bewan ATM/PCI "st" card on a P166 machine.

Thanks to Dave for his patience in testing!

Comments on this section should be addressed to its maintainer: Jacques Nilo <jnilo@users.sourceforge.net>.

5.2. Step 1: declare the pppatm package

In order to be able to get connected through PPPoA you will need a special version of ppp patched for PPPoA support. This support is provided by a pppoatm.so "plugin" which is − unfortunately − only available for ppp version 2.4.0b2 (The "standard" Bering ppp version is 2.4.1). The pppatm.lrp package is nothing more than this patched version of ppp 2.4.0b2 which was developped by Michael Mitchell. This package will replace the ppp.lrp package provided on your Bering floppy.

Note: pppd will appear as 2.4.0b1 in syslog but it's really pppd 2.4.0b2 !

Boot your Bering floppy image. Once the LEAF menu appears get access to the linux shell by (q)uitting the menu. Edit the `syslinux.cfg` file and REPLACE the pump entry by pppatm in the LRP= list of packages to be loaded at boot. Check the Bering installation guide to learn how to do that.

Your `syslinux.cfg` file will then look like (adjust to your tastes):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,pppatm,keyboard,shorwall,dnscache,webl
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

The pppatm.lrp package is available here.

5.3. Step 2: declare the ppp and pppoatm modules

In order to have a PPPoA connection working, you need to have both ppp and pppoatm support enabled through the appropriate kernel modules. You also need to declare the driver(s) module(s) of your network card(s). In the following example, we assume that the external connection to the Internet is provided by a Bewan ATM/PCI card while the internal network goes through a standard ne 2000 PCI card.

All the modules which are necessary for ppp support are provided on the standard Bering floppy. You just need to "declare" them since they are not loaded by default. As far as the pppoatm module is concerned you will have to download it from the Bering modules download area and store it in `/lib/modules`.

The module drivers for the Bewan ATM/PCI card are provided in the driver contrib section. Store them in `/lib/modules` as well. Other ATM drivers are available here.

To declare your modules, go to the LEAF Packages configuration menu and choose modules. Enter 1) to edit the `/etc/modules` file and enter the following information:

```
# 8390 based ethernet cards
8390
ne2k-pci

# Modules needed for PPP connection
slhc
ppp_generic

# PPPOA support
pppoatm

# Bewan ATM-PCI "st" drivers
unicorn_atm
unicorn_pci ActivationMode=1

# Masquerading 'helper' modules
ip_conntrack_ftp
ip_conntrack_irc
ip_nat_ftp
ip_nat_irc
```

Backup the modules.lrp package.

5.4. Step 3: configure pppatm

Connection with your ISP will be handled by PPP. The PPP Howto document will give you very detailed information about this protocol and how to set-up its numerous parameters.

Through the LEAF packages configuration menu get access to pppatm configuration. The following menu will show-up:

```
                        pppatm configuration files

       1) ISP pppd options
       2) System wide pppd options
       3) chap secret
       4) pap secret
```

```
       5) pppd daemon script

  q) quit
  ------------------------------------------------------------------------------
       Selection:
```

Enter 1) and adjust the corresponding `/etc/ppp/peers/dsl-provider` file:

```
#
# Adjust here VP/VC - depends on country & ISP
# UK/BT: 0.38 - US/BE/FR: 8.35
#
plugin /usr/lib/pppd/pppoatm.so 0.38
#
# If chap or pap identification uncomment the #name "ISPUserID" line
# and replace ISPUserID with your ISP user name
# There should be a matching entry in /etc/ppp/pap-secrets or chap-secrets
#
#name "ISPUserID"
lock
noipdefault
noauth
defaultroute
hide-password
lcp-echo-interval 20
lcp-echo-failure 3
maxfail 0
persist
```

The most important parameters in this file are the VP.VC combination which depends on your country and/or your ISP and the name parameter.

You should not need to adjust 2).

Edit either the CHAP (Entry 3) or PAP (Entry 4) option to set up how your system authenticates. If you edit chap, replace #ISPUserID and ISPUserPassword this the relevant information.

```
# Secrets for authentication using CHAP
# client        server  secret                  IP addresses
#ISPUserID      *       ISPUserPassword
```

ISPUserID must exactly match the entry that you made for the name parameter in Entry 1) "ISP pppd options" file. The "*" can be replaced with the IP address or name of the server you are dialling into if you know it. Usually, an asterisk is sufficient.

If you want to authenticate using PAP, add the same entry to the PAP item instead.

Backup the pppatm.lrp package.

5.5. Step 4: configure your interfaces file

Trough the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo ppp0 eth0

iface lo inet loopback

iface ppp0 inet ppp
        provider dsl-provider

iface eth0 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
```

In this `/etc/network/interfaces` file the lo, ppp0 and eth0 interfaces are brought up automatically when the *ifup -a* statement is executed at boot time by the `/etc/init.d/networking` script.

The "iface ppp0 inet ppp" section defines the external address of the router and activates the `pon` script

The "iface eth0 inet static" defines the internal address of the router.

Backup the etc.lrp package.

5.6. Step 5: configure Shorewall

Through the LEAF packages configuration menu, choose shorwall and check the three following files:

A/ The `interfaces` file (entry 3) defines your interfaces. Here connection to the net goes through ppp0. So we must set:

```
(...)
#ZONE   INTERFACE       BROADCAST       OPTIONS
net     ppp0            -
loc     eth0            detect          routestopped
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Do not forget the "-" under the BROADCAST heading for the net/ppp0 entry.

B/ The `masq` file (entry 7). It should look like:

```
(...)
#INTERFACE              SUBNET
ppp0                    eth0
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Backup the shorwall.lrp package.

5.7. Step 7: reboot...

Your PPPoA connection should be established automatically. Type *plog* to check the login sequence with your ISP. If there is no output check `/var/log/syslog` to get a clue on potential problems.

6. PPTP/PPPoA configuration

6.1. Objectives

We assume here that you want to connect your LEAF router to the Internet via an Alcatel SpeedTouch home ADSL modem which supports both PPPoE and PPPoA connections. The PPPoE connection is covered in another section. For the PPPoA connection, we assume that your modem is connected to a dedicated NIC as eth0 and will communicate with your router through the pptp protocol. What is described here corresponds to section 3.2.5 of the DSL How−To document. The traffic to your internal network goes through eth0 while access to the Internet via PPPoA goes through ppp0.

The PPP−Howto, the PPTP−Client project and the DSL−Howto are two useful references for this section.

Comments on this section should be addressed to its maintainer: Jacques Nilo <jnilo@users.sourceforge.net>.

6.2. Step 1: declare the ppp and the pptp packages

Boot a Bering floppy image. Once the LEAF menu appears get access to the linux shell by (q)uitting the menu. Edit the `syslinux.cfg` file and REPLACE the pump entry by ppp,pptp in the LRP= list of packages to be loaded at boot. Check the Bering installation guide to learn how to do that.

Your `syslinux.cfg` file will then look like (adjust to your tastes):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,ppp,pptp,keyboard,shorwall,dnscache,we
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

The ppp package is provided on the standard Bering floppy. The pptp.lrp package is available here. Check the Bering installation guide.

6.3. Step 2: declare the ppp modules

In order to have a PPTP/PPPoA connection working, you need to have ppp support enabled through the appropriate kernel modules. You also need to declare the driver(s) module(s) of your network card(s). In the following example, we assume that both ethernet interfaces are provided through a standard ne 2000 PCI card.

All the modules which are necessary for a PPTP/PPPoA connection are provided on the standard Bering floppy. You just need to "declare" them since they are not loaded by default. As far as your network cards are concerned, the most popular driver modules are provided in `/lib/modules` but you might need to download the one corresponding to your own hardware from the Bering modules download area. Refer to the Bering installation guide to learn how to do that.

To declare your modules, go to the LEAF Packages configuration menu and choose modules. Enter 1) to edit the `/etc/modules` file and enter the following information:

```
# 8390 based ethernet cards
8390
ne2k-pci

# Modules needed for PPTP/PPPoA connection
slhc
n_hdlc
ppp_generic
ppp_async

# Masquerading 'helper' modules
ip_conntrack_ftp
ip_conntrack_irc
ip_nat_ftp
ip_nat_irc
```

The `/etc/modules` file provided in the Bering distro is already setup with those entries commented out. Just remove the leading # sign to activate the corresponding module.

Backup the modules.lrp package.

6.4. Step 3: configure ppp

Connection with your ISP will be handled by PPP. The PPP Howto document will give you very detailed information about this protocol and how to set-up its numerous parameters.

Through the LEAF packages configuration menu get access to ppp configuration. The following menu will show-up

```
                        ppp configuration files

       1) ISP pppd options
       2) ISP login script
       3) System wide pppd options
       4) chap secret
       5) pap secret
       6) pppd daemon script

  q) quit
  --------------------------------------------------------------------------
       Selection:
```

Enter 1) and 2) and empty out the corresponding files completely

Enter 3) allows you to adjust the parameter of your ppp connection through the `/etc/ppp/options` file. This file must contain:

```
debug
name "ISPUserID"
noauth
noipdefault
defaulroute
```

Edit either the CHAP (Entry 4) or PAP (Entry 5) option to set up how your system authenticates.

For PAP authentication, choose the PAP option and add a line saying "<ISPUserID> * <ISPUserPassword> to the bottom of the file. <ISPUserID> is the same entry that you made in Entry 3) − the "System wide pppd options" file. The <ISPUserPassword> entry is self−explanatory. The "*" can be replaced with the IP address or name of the server you are dialling into if you know it. Usually, an asterisk is sufficient.

If you want to authenticate using CHAP, add the same entry to the CHAP item instead.

Backup the ppp.lrp package.

6.5. Step 4: configure your interfaces file

Trough the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo eth0 eth1

iface lo inet loopback

iface eth0 inet static
        address 10.0.0.1
        masklen 24
        broadcast 10.0.0.255
        up pptp 10.0.0.138

iface eth1 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
```

In this `/etc/network/interfaces` file the lo, eth0 and eth1 interfaces are brought up automatically when the *ifup −a* statement is executed at boot time by the `/etc/init.d/networking` script.

The "iface eth0 inet static" section defines the external address of the router and says:

- Bring up eth0 at address 10.0.0.1
- Execute the `pptp 10.0.0.138` command once eth0 is up to establish the PPTP/PPPoA connection.

The "iface eth1 inet static" defines the internal address of the router.

Backup the etc.lrp package.

6.6. Step 5: configure Shorewall

Through the LEAF packages configuration menu, choose shorwall and check the three following files:

A/ The `interfaces` file (entry 3) defines your interfaces. Here connection to the net goes through ppp0. So we must set:

```
(...)
```

```
#ZONE   INTERFACE       BROADCAST       OPTIONS
net     ppp0            -               routefilter
loc     eth1            detect          routestopped
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Do not forget the "−" under the BROADCAST heading for the net/ppp0 entry.

B/ The `masq` file (entry 7). With a dial−up modem setup it should look like:

```
(...)
#INTERFACE              SUBNET
ppp0                    eth1
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

C/ You may also need to edit the `config` file (entry 12) to adjust the CLAMPMSS variable to "yes":

```
(...)
# Set this variable to "Yes" or "yes" if you want the TCP "Clamp MSS to PMTU"
# option. This option is most commonly required when your internet
# interface is some variant of PPP (PPTP or PPPoE). Your kernel must
#
# If left blank, or set to "No" or "no", the option is not enabled.
#
CLAMPMSS="yes"
(...)
```

Backup the shorwall.lrp package.

6.7. Step 7: reboot...

Your modem connection should be established automatically. Type *plog* to check the login sequence with your ISP. If there is no output check `/var/log/syslog` to get a clue on potential problems.

Prev PPPoA configuration | Home | Up | Next ISDN Configuration

7. ISDN Configuration

7.1. Objectives

We assume here that you want to make a connection to the internet using synchroneous ppp and that you use an internal passive ISDN card. The isdn4linux driver is documented for Euro ISDN

The setup is tested with different providers (T−online, MSN, and my own ppp Server) on 486er and Pentium machines using the AVM Fritz and an Elsa Microlink card.

Specific questions conserning the Hisax driver can be looked up at the i4l faq and the Readme for Hisax

Special cases like canal bundeling, callback etc are not yet tested but should be possible. Refer to the hisax guide. Although the use of active cards, like the avm b1, is possible,it is not tested, and should require some extra work.

Users with external ISDN modems should look at the ppp dial up page..

Comments on this section should be addressed to its maintainer: Eric Wolzak <leaf@wolzak.de>.

ISDN−cards connect quiet, and usually without any signs. If you have a wrongly configured machine on your network, you could experience a lot of undesired connections: you will not notice it until the next telephone bill!. So, especially for a start, check your messages file regulary. I use the beep.lrp which gives an audible signal on connecting. During the setup disconnect the ISDNline until you know that all other parts do function.

7.2. Step 1: Download and declare the isdn.lrp package

Download the isdn.lrp package from Eric's site and store it on your Bering disquette. If you need space to do that, refer to the installation guide to learn how to do that.

Boot your Bering floppy image. Once the LEAF menu appears get access to the linux shell by (q)uitting the menu. Edit the `syslinux.cfg` file and REPLACE the pump entry by isdn in the LRP= list of packages to be loaded at boot. Check the Bering installation guide to learn how to do that.

Your `syslinux.cfg` file will then look like (adjust to your tastes):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,keyboard,isdn,shorwall,dnscache,weblet
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

7.3. Step 2: download the isdn.o and the appropriate hisax.o modules

It's now time to download ISDN modules. You need both the isdn.o module and the hisax.o modules.

You can use the "normal" hisax.o module, with built−in support for every cards. But this module has a size of about 600K, so you will have a problem getting everything on a single disk. This is why I compiled "partial" hisax modules, each supporting a small group of cards. It will be a little be more work to select the correct one if you have an exotic card, but the size of less than 250 K will be worth the trouble. To see what Hisax module you need check the following table

Table 1. Available ISDN modules

Model	Brand	Type	Options
Teles_16_0	Teles S0−16.0	1	irq, mem, io
Teles_16_0	Teles S−8 en compatibele	2	irq, mem
Teles_16_3	Teles S0−16.3	3	irq, io
Teles_16_3	Teles/Creatix PNP	4	irq, io0 (ISAC), io1 (HSCX)
Teles PCMCIA	Teles_PCMCIA	8	irq, io
TelesPCI	Teles PCi	21	no parameter
S0_Box	Teles/Creatix parallel port S0Box	25	irq, io (of the used lpt port)
hisax_AVM_A1	AVM A1 (Fritz)	5	irq, io
hisax_AVM_A1	Teledat 150	5	irq,io
hisax_FRITZ_PCI	AVM Fritz!PnP	27	irq, io (from isapnp setup)
hisax_FRITZ_PCI	AVM Fritz!PCI	27	no parameter
hisax_AVM_A1_PCMCIA	AVM A1 "Fritz!PCMCIA"	26	irq, io (set with card manager)
hisax_ELSA	Elsa Microlink ISA	6	io or nothing for autodetect (the iobase is required only if you have more than one ELSA card in your PC)
hisax_ELSA	Elsa Quickstep series isa	7	irq, io (from isapnp setup)
hisax_ELSA	Elsa Quickstep 1000 pci	18	none
hisax_ELSA	Elsa Quickstep 3000 pci	18	none
hisax_ELSA	Elsa PCMCIA	10	irq, io (set with card manager)
hisax_IX1MICROR2	ITK ix1−micro Revision 2 card	9	irq, io
hisax_DIEHLDIVA	Eicon Diehl Diva none Pro version	11	irq, io
hisax_ASUSCOM	AsusCom ISA (isdnlink)	12	irq, io (from isapnp setup)
hisax_ASUSCOM	Dynalink IS64PH (oem)	12	irq, io (from isapnp setup)
hisax_ASUSCOM	PCBit−DP (oem)	12	irq, io (from isapnp setup)
hisax_TELEINT	TELEINT SA1 semiactiv	13	irq,io
hisax_HFCS	HFC−S 2BDS0 based cards	13	irq, io

hisax_HFCS	teles 16.3c pnp	14	irq, io
hisax_SEDLBAUER	Sedlbauer Speed Card	15	irq, io
hisax_SEDLBAUER	Sedlbauer PC/104	15	irq,io
hisax_SEDLBAUER	Sedlbbauer pci	15	none
?	Acer	30	irq, io (from isapnp setup)
hisax_SPORTSTER	Stollmann tina−pp V3	16	irq,io
hisax_SPORTSTER	USR Sportster internal TA	16	irq, io
hisax_MIC	ITH MIC Card	17	irq, io
hisax_NETJET	Ovislink ISDN sc100−p card	20	none
hisax_NETJET	Traverse Technologies Netjet	20	none
hisax_NETJET_U	Traverse Technologies Netspider U	38	none
hisax_NICCY	Dr. Neuhaus Niccy PNP	24	irq, io0, io1 (from isapnp setup)
hisax_NICCY	Dr. Neuhaus Niccy PCI	24	no parameter
hisax_ISURF	Siemens I_Talk (IsAR chip)	29 ?	irq, io, memory (from isapnp setup)
hisax_ISURF	Siemens I−Surf (ISAR chip) 1.0	29	irq, io, memory (from isapnp setup)
hisax_ASUSCOM	Siemens I−Surf (IPAC chip) 2.0	12	irq, io (from isapnp setup)
hisax_HSTSAPHIR	HST Saphir card	31	irq, io
hisax_BKM_A4T	Berkom Telekom A4T Card	32	none
hisax_SCT_QUADRO	Scitel Quadro card	33	subcontroller (4*S0, subctrl 1...4)
hisax_GAZEL	Gazel card isa	34	irq,io
hisax_GAZEL	Gazel card pci	34	none
hisax_HFC_PCI	HFC−S PCI 2BDS0	35	none
hisax_W6692	Winbond W6692 based PCI cards	36	none
hisax_HFC_SX	HFC−S+	37	irq, io
hisax_HFC_SX	HFC−SP	37	irq, io
hisax_HFC_SX	HFC−SP/PCMCIA	39	irq,io (set with cardmgr)

Once you have downloaded the appropriate module rename it to hisax.o and copy it to the `/lib/modules` directory.

7.4. Step 3: declare the ISDN modules

In order to have an ISDN connection working, you need to have ISDN support enabled through the appropriate kernel modules. You also need to declare the driver(s) module(s) of your internal network card(s). In the following example, we assume that your internal network card is a NE2000 PCI.

To declare your modules, go to the LEAF Packages configuration menu and choose modules. Enter 1) to edit the `/etc/modules` file and enter the following information:

```
# 8390 based ethernet cards
8390
ne2k-pci

# Modules needed for ISDN
# Look for type, io and irq settings at help page of isdn.lrp documentation
slhc
isdn
hisax type=   io= irq=

# Masquerading 'helper' modules
ip_conntrack_ftp
ip_conntrack_irc
ip_nat_ftp
ip_nat_irc
```

the paramater you have to give for a certain card are listed in the table above

as an example: to use Fritz card from AVM(A1) download the module hisax_AVM_A1. rename it to hisax.o. if you fritzcard is configured with irq 7 and ioport 330 you enter hisax type=5 irq=7 io=0x330

Backup the modules.lrp package.

7.5. Step 4: configure ISDN

Most options are already defined with reasonable default values. But some settings must be defined in every case. If you have a static ip number you should also change the according parameter.

Through the LEAF Package configuration menu choose isdn. The following menu will appear:

```
                        isdn configuration files

       1) ipppd options
       2) password und userid
       3) ipppd scipts to startup the ipppd interfaces
```

select 1 now you edit the User setting and enter here the name or number that you need to identify yourself

```
(---)
#  USER Dependent options
#
USER="eric@foobar.com"
#
# your MSN depending on your country without areaprefix
#
MYMSN=
#
# Provider MSN
#
REMMSN=
#
# Hangup after idletime in seconds 0 for no hangup
#
TIMEOUT=60
#
(---)
```

What your MSN is, is depending on the country you live in.If you are in doubt, ask your local telco.For a few countries you can find the answer here. The remote MSN (REMMSN) is the nummer you have to dial from the connection the router is attached to, including extra digits exactly as you would dial it.

You might want to change the time set to keep the line up if there is no activity. As a default it is set to 60 (sec),which is relatively short. You change this with the parameter TIMEOUT

now use the password and userid from the isdn configuration menu set your login name (eric@foobar).I can login with this name on any computer (*). I have to identify me with the password ("this_is_a_secret")

```
# This is a pap-secrets file
#
#papname * papsecret
eric@foobar.com *  "this_is_a_secret"
```

If you have ppp installed, the pap−secrets file is shared and this could give problems with the backup. You don't need ppp for isdn.lrp

backup the isdn package

7.6. Step 5: configure your interfaces file

Through the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo eth0

iface lo inet loopback

iface eth0 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
```

attention the internal interface is now eth0 assuming you have only one interface!

your external interface is now ippp0, but this interface is not setup in the interfaces file

7.7. Step 6: configure Shorewall

Through the LEAF packages configuration menu, choose shorwall and check the two following files:

A/ The `interfaces` file (entry 3) defines your interfaces. Here connection to the net goes through ippp0 and the connection to the internal network through eth0. So we must set:

```
(...)
#ZONE   INTERFACE       BROADCAST       OPTIONS
```

```
net     ippp0           -               dhcp,routefilter,norfc1918
loc     eth0            detect          routestopped
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Do not forget the "−" under the BROADCAST heading for the net/ippp0 entry.

B/ The `masq` file (entry 7). In this type of setting it should look like:

```
(...)
#INTERFACE              SUBNET
ippp0                   eth0
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Backup the shorwall.lrp package.

Prev
PPTP/PPPoA configuration

Home
Up

Next
Creating a bootable Bering CD−ROM

8. Creating a bootable Bering CD−ROM

8.1. Objectives

These instructions assume that you already have some knowledge of Bering and a working distribution running out of one or two floppies.

They have been tested with Bering v1.0−rc1 on several hardware configurations, using only IDE CD−ROM's. No SCSI support is planned at this stage.

Comments on this section should be addressed to its maintainer: Luis Correia <lfcorreia@users.sourceforge.net>.

8.2. Step 1: preparing the distro

First make sure you have your Bering floppy distro already working. You may want to take one or more of the following actions:

- Define root password
- Generate the ssh keys, if you will use them

Make sure to read the CD−Rom section of Bering user's guide chapter on "Booting Bering from different boot−media".

8.3. Step 2: downloading the required packages

From now on we are going to use a Windows machine to create the CD. Linux users should have no problem in following.

Download the two following packages, syslinux and cdrtools. Unpack them with WinZip. Also download the makeiso.bat MS−DOS bat file from the Bering "contrib" directory.

Create a new directory. It can be anywhere but in practice I recommend to create it next to the root (e.g. C:\) since it will be easier to access it at a later stage from the DOS prompt. Let's call it BCD ("Bering CD"). We will have the following directory hierarchy:

```
C:\
 |
  --- C:\BCD
       |
        --- C:\BCD\diskcontent
```

Once this is done put in the C:\BCD directory the following 3 files:

- `mkisofs.exe` (1) and `cygwin1.dll` (2) extracted out of the cdrtool directory.
- `makeiso.bat` (3) downloaded from the Bering "contrib" directory.

Then put in the C:\BCD\diskcontent directory the following file:

- `isolinux.bin` extracted out of the syslinux directory.

Other versions of these files − older or more recent ones − may also work. You are on your own here!

8.4. Step 3: preparing the Bering CD−Rom content

If you are in a hurry, download the initrd.cdrom file from the Bering "contrib" directory and rename it initrd.lrp. This is an initrd.lrp "package" which includes the necessary kernel modules to access a CD−Rom at boot time. You can also create it yourself as follow:

Boot your working Bering floppy. In the `/boot/lib/modules` directory put the following modules that will allow boot time CD−Rom support (those modules can be found in the Bering modules download area):

```
cdrom.o
ide-mod.o
ide-cd.o
ide-probe-mod.o
isofs.o
```

Declare those names, without the ".o" suffix in the `/boot/etc/modules` file through the initrd package menu. The order MUST be respected.

Now backup the initrd.lrp package !

Copy all the files from your working Bering floppy to the `C:\BCD\diskcontent` directory (except initrd.lrp if you have not created it yourself as described above, in which case you will put in the C:\BCD dir the one you downloaded).

In this directory do the following:

- Rename `syslinux.cfg` to `isolinux.cfg`
- Delete `ldlinux.sys`
- Edit `isolinux.cfg` and replace the /dev/fd0u1680 entries (after boot= and PKGPATH=) by /dev/cdrom.
- Add any package you might need out of the CD. Do not forget the hackers though...

After that your `isolinux.cfg` file will look like:

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/cdrom:iso9660 PKGPATH=/c
   LRP=root,etc,local,modules,ppp,pppoe,keyboard,shorwall,dnscache,weblet,libz,routerst,sshd,ti
```

8.5. Step 4: making the CD

Get access to MS−DOS from within Windows. Change your directory to C:\BCD. Then execute the makeiso command file. This file contains one single line with the following command:

```
mkisofs -o bering.iso -b isolinux.bin -c isolinux.cat -no-emul-boot -boot-load-size 4 -boot-inf
```

You should now have a `bering.iso` CD−Rom image in `C:\BCD`. You can now create your Bering CD from this ISO image with your favorite CD burner program.

Change your BIOS settings to declare your CD−Rom as the first boot device. You should be all set.

It is of course much better to test your Bering CD with a CD−RW :−). But be aware that a lot of "old" CD drives just won't be able to read them. The CD−RW will be used for testing on a recent machine. Once you are happy with your image you will have to burn a traditionnal CD−Rom for your old i486−based Bering router :−)

8.6. Support

I read both the leaf−user and the leaf−devel lists. You may put your questions there.

8.7. Thanks to...

Charles Steinkuehler for creating the *stein series,

Jacques Nilo & Eric Wolzak for the Bering series,

Allen Hillery for the hints & Christian Hostelet for beta testing,

all LEAF developpers,

my friend João Alves for his helpful linux support and

Mike Noyes for keeping up his excellent work on the LEAF site.

9. Booting Bering from different boot−media

9.1. Objectives

These instructions are for those who want to boot Bering from something else that the traditional single floppy setup. We assume that you already have some knowledge of Bering.

Many thanks to Allen Hillery for his contribution to this section !

Comments on this section should be addressed to its maintainers: Jacques Nilo <jnilo@users.sourceforge.net> or Eric Wolzak <leaf@wolzak.de>.

9.2. The single floppy drive setup

The poor man setup ... Do not worry you can still do many things. Here are the tricks:

The main problem when you have got a single floppy drive is space. Especially if you are willing to use those big fat packages like sshd.lrp or ipsec.lrp. But you can still use them in such an environnement. There are basically two approaches:

The first one is to remove useless components from the Bering floppy. Refer to the installation guide to learn how to do that.

But most of the time, for big applications, one floppy won't fit. You then have to setup your distro on two floppies, while still using a single drive. The strategy is as follow:

On the first floppy keep only the following files: `linux`, `ldlinux.sys`, `syslinux.dpy`, `syslinux.cfg` and `initrd.lrp`.

On the second floppy put all the remaining LEAF packages that you will need. You have a full floppy available! It can be 1440k, 1680k or 1723k formatted but it should be the same format for both floppies. 1680k is generally working without any problem and is a de facto LEAF standard.

Then edit the `syslinux.cfg` file of the first floppy. You will enter something like:

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos diskwait=
   LRP=root,etc,local,modules,pump,keyboard,shorwall,dnscache,sshd,ipsec,weblet
```

The last two lines ("default linux ... ipsec,weblet") must be typed as a single one in `syslinux.cfg`

Notice the *diskwait=yes* statement. Once the first floppy will be booted, Bering will ask you to enter the second floppy and will then load the *.lrp LEAF packages.

In this setup you can leave the second floppy on your drive if you want to make changes to your configuration files and backup the corresponding packages. What is on the first floppy generally do not need to be

backed−up !

You can optimize this setup by declaring all your modules in `/boot/etc/modules`, moving them from `/lib/modules` to `/boot/lib/modules`. Then backup initrd.lrp. Then you won't need modules.lrp anymore since everything will be stored in initrd.lrp on the first floppy :−)

9.3. The two−floppy drives setup

Here we assume that you have two floppy drives available, namely fd0u1680 and fd1u1680 (assuming 1680k formatted floppies). The first floppy will be s standard Bering floppy. The second one will only contain *.lrp LEAF packages that do not fit on the first floppy. In this setup *.lrp LEAF packages can be on any disk and you only have to adjust the PKGPATH statement of the first "booting" floppy:

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos PKGPATH=/
   LRP=root,etc,local,modules,pump,keyboard,shorwall,dnscache,sshd,ipsec,weblet
```

9.4. Booting from an IDE device

To install Bering on an IDE device, proceed as follow:

Make sure your IDE device has a first bootable partition and is DOS formatted. With the Windows rescue disk you will have the fdisk and the format utilities to help you doing that. With a linux rescue floppy disk, fdisk and mkfsdos will be your friends.

Be careful: you will be destroying any pre−existing data !

Once you hard disk is formatted install syslinux. You can install syslinux either from a windows or a linux rescue floppy. Boot your floppy then issue the following command:

```
syslinux [−s] /dev/hda1
```

The −s flag might be required for syslinux to work with old buggy BIOSes. See the syslinux web site for more instructions.

Boot a Bering floppy. Install on the /boot/lib/modules the ide−mod.o, ide−disk.o and the ide−probe−mod.o modules. Then declare those modules in /boot/etc/modules through the initrd package configuration menu in this order. Then backup the initrd.lrp package. Once this is done, edit the `syslinux.cfg` file which will look like:

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/hda1:msdos PKGPATH=/dev/
   LRP=root,etc,local,modules,pump,keyboard,shorwall,dnscache,sshd,ipsec,weblet
```

Once you have finished with your floppy preparation, copy its content to the IDE device that you prepared earlier. You should now be able to boot from the IDE device.

9.5. Booting from a CD−Rom with isolinux

This section does not cover the creation of the Bering cd−rom which is explained in a separate section.

The start options for isolinux are similar to the syslinux options. By default they look like this:

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/cdrom:iso9660 PKGPATH=/c
```

There are some important differences with the syslinux parameters:

- The boot filesystem − iso9660 − must be specified after the boot device (/dev/cdrom) in the boot= statement. Use a colon (:) as a delimiter
- The devices in your PKGPATH= statement can be given different filesystems The former are seperated from the later with a colon(:)
- The order in which the devices (/dev/cdrom, /dev/fd0) are declared in the PKGPATH= statement is important. Packages will be picked−up in this order which means that you can override a package from the CD−rom with one provided on the floppy

If you are booting from a CD−Rom the list of packages in the LRP= statement might be pretty long. The problem is that there is a limit to the lenght of isolinux.cfg statements which cannot exceed 255 characters. To avoid this limitation, you can declare the list of packages you are going to use in a file called `lrpkg.cfg`. When this file exists on the boot device, the package list will be read from it.

This file consists of a single record with a list of packages separated by commas. It looks like:

```
# cat lrpkg.cfg

root,etc,local,modules,pump,keyboard,shorwall,dnscache,weblet
```

This file can be present in more locations. The last location in the PKGPATH= statement will be used. So you have the possibility to have a "standard" lrpkg.cfg on your CD and for special occasions or testing, you can have another one on a floppy.

As stated before you can load a package stored on different devices. This is useful in the following situations:

- To have access to an updated package on the floppy
- To do a partial backup of of a package on the floppy. Especially useful for configuration files ! Have you ever tried to backup a package on a CD−Rom ? :−)
- To do testing

You can indicate for every package where to look first: R(everse),r or F(orward),f

With the F(orward) option the searching for the package starts on the left in the package path.

- With the uppercase F it stops as soon as the first occurence of the package is found.
- With the lowercase f the search start from left to right but all occurences of the packages are loaded. This option is taken to load a partial backup. Be sure that the package found first is the one with the standard configuration. The one found in the second place will overwrite the saved files with the

individual options.

The same rules applies for the R(everse) option. Especially the Uppercase R can be used to load a complete new version of a package

The full syntax for the package list is *package_name:option,package_name:option,...*

```
#cat lrpkg.cfg
root:F,etc:f,local:R,modules:R,pump:r
```

Example. Let's assume you have the following setup in your `isolinux.cfg` file.

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/cdrom:iso9660 PKGPATH=/d
```

and the following `lrpkg.cfg` package file.

```
root,etc,local,modules,pump:f,keyboard,shorwall:r,dnscache,weblet
```

The search order for pump (f) will be: cd −−> floppy. To load pump only from floppy use R.

The search order for shorwall (r) will be: floppy−−> cd. To load shorwall only from boot−cd use F.

You will be able to see the search order at boot time.

9.6. Partial backup of packages to/from floppy

Saving a partial backup to floppy disk

- If you want to backup parts of the package that are not in the /etc and /var/lib/lprkg directories, you have to be sure there is a /var/lib/lrpkg/PACKAGE.local file for each PACKAGE.lrp you are doing partial backup of. This file contains the list of files to be saved in the partial backup. See doc. for format. This list should include local configuration files and any binary files that have been updated. (always include "I /var/lib/lrpkg/PACKAGE.*" to save the *.local files in the Partial backup of PACKAGE.lrp)
- Set back up to partial (p #) and set the backup device (d #) to something like fd0 and msdos

Loading partial backup from floppy disk after booting cdrom

- Check syslinux.cfg on boot cd to see if PKGPATH includes partial backup device the default is PKGPATH=/dev/cdrom:iso9660,/dev/fd0:msdos
- set the load order in lrpkg.cfg file on the floppy disk to load CDROM version of the package then the floppy version of the partial back of the package.

This ":f" (the default) will first load the cdrom version then the floppy updates it they exist.

Use ":R" to load the floppy version a full package and totally avoid the cdrom version of the package.

You can NOT do a partial back up of initrd.lrp because it loaded directly off the boot disk. If the version on the cdrom needs to be change you must make a new cdrom or use a boot floppy disk with a new initrd.lrp and then you can load other packages off the cdrom.

Prev
Creating a bootable Bering CD−ROM

Home
Up

Next
Installing and booting Bering from a M−Systems DiskOnChip

10. Installing and booting Bering from a M−Systems DiskOnChip

10.1. Objectives

These instructions describe how to modify a stock Bering floppy disk image to run from a M−Systems DiskOnChip. They were tested using Bering v1.0−rc3 on an Advantech PCA−6145B single board computer with a 4 MB DiskOnChip 2000. It is assumed that you have the ability to boot your DoC−enabled device from a floppy drive during setup.

Comments on this section should be sent to Brad Fritz at <bradfritz@users.sourceforge.net>. This is revision $Revision: 1.1 $. Please include the revision number with any comments.

10.2. Step 1: prepare the boot floppy

Obtain a working Bering v1.0−rc3 or newer boot floppy and perform the following steps:

a. Download the appropriate MTD modules for your DoC from the `drivers/mtd` directory of the Bering modules tree. For DiskOnChip 2000 products, you will need `mtdcore.o`, `docecc.o`, `doc2000.o`, `docprobe.o` and `nftl.o`. The `docecc.o`, `doc2000.o`, and `docprobe.o` modules are in the `drivers/mtd/devices` subdirectory.
b. Download a fdisk package (or equivalent) that contains the **mkfs.msdos**, **fdisk** and **syslinux** commands.
c. Copy the modules and the fdisk package to your Bering floppy disk. If there is not enough room, you can delete Bering packages that you do not need or use a second MS−DOS formatted floppy disk.

10.3. Step 2: apply bug fixes

Bering rc3 contains two bugs in initrd.lrp that that need to be fixed before booting from a DoC will work properly. A typographical error in `/var/lib/lrpkg/root.dev.mk` causes the `/dev/nftl*` devices to have an incorrect major number. A modification of `/var/lib/lrpkg/root.linuxrc` will prevent `/dev/nftla1` from being mounted twice and causing the DoC boot to hang.

a. Boot the floppy you prepared in the previous step.
b. Fix the nftla device major numbers by changing line 31 in `/var/lib/lrpkg/root.dev.mk` from:

```
#Disk−On−Chip
makedevs nftla b 3 0 0 4 s >null 2>&1
```

to:

```
#Disk−On−Chip
makedevs nftla b 93 0 0 4 s >null 2>&1
```

to change the major number from 3 to 93.

c. Fix the already created nftla devices by running:

```
rm /dev/nftla*
makedevs /dev/nftla b 93 0 0 4 s
```

from the command prompt.

d. Fix the double mounting problem by adjusting `/var/lib/lrpkg/root.linuxrc`. Add an else block at line 246 (assuming you are using Bering rc3). Lines 246 through 249 in the snippet below are the inserted lines.

```
244     done
245     IFS=$OIFS
246 else
247     bootfs=`cat /var/lib/lrpkg/boot.fstype`
248     rdevlist="/dev/boot:+$MNT:-$bootfs,$rdevlist"
249     devlist="$devlist,/dev/boot:+$MNT:-$bootfs"
250 fi
```

10.4. Step 3: configure Bering for DoC booting

a. Mount the floppy disk, move the DoC modules to the `/boot/lib/modules` directory, and stage the fdisk package.

```
mount -t msdos /dev/fd0u1680 /mnt
cd /mnt
mv mtdcore.o docecc.o doc2000.o docprobe.o nftl.o /boot/lib/modules
mv fdisk.lrp /tmp
cd /
umount /mnt
```

b. Edit `/boot/etc/modules` and add the following lines. The order of the lines is *very* important.

```
mtdcore
docecc
doc2000
docprobe
nftl
```

It is a good idea to make sure there is a blank line at the end of the `/boot/etc/modules` file.

c. Backup the initrd package.

If you do not backup initrd, your changes will not be transferred to the DoC in step 4.

10.5. Step 4: prepare the DoC

a. Load the MTD modules:

```
cd /boot/lib/modules
insmod ./mtdcore.o
```

```
insmod ./docecc.o
insmod ./doc2000.o
insmod ./docprobe.o
insmod ./nftl.o
```

b. After **insmod**ing the `docprobe.o` module, you should see output that looks similar to:

```
Possible DiskOnChip with unknown ChipID FF found at 0xc8000
Possible DiskOnChip with unknown ChipID FF found at 0xca000
Possible DiskOnChip with unknown ChipID FF found at 0xcc000
 [..]
DiskOnChip 2000 found at address 0xD8000
Ignoring DiskOnChip 2000 at 0xDA000 − already configured
Ignoring DiskOnChip 2000 at 0xDC000 − already configured
Ignoring DiskOnChip 2000 at 0xDE000 − already configured
Possible DiskOnChip with unknown ChipID FF found at 0xe0000
Possible DiskOnChip with unknown ChipID FF found at 0xe2000
 [..]
```

c. Verify the DoC has been recognized by running **cat /proc/mtd**. The output should look similar to:

```
dev:    size   erasesize  name
mtd0: 00400000 00002000 "DiskOnChip 2000"
```

d. Install the fdisk package:

```
cd /tmp
lrpkg −i fdisk
```

e. Partition the DoC. Run **fdisk /dev/nftla** and create a single DOS 12−bit FAT partition and set it to active. The hex code for DOS 12−bit FAT is 0x1.

f. Create an MS−DOS filesystem on the DoC by running **mkfs.msdos /dev/nftla1**.

g. Mount the newly created filesystem and copy the Bering files to it.

```
mkdir /doc
mount −t msdos /dev/nftla1 /doc
mount −t msdos /dev/fd0u1680 /mnt
cp /mnt/* /doc
umount /mnt
```

h. Edit the DoC `/doc/syslinux.cfg` file and remove the **PKGPATH=/dev/fd0u1680** parameter and change the boot parameter **boot=/dev/nftla1:msdos**.

i. Unmount the DoC partition with **umount /doc**.

j. Make the DoC bootable by running **syslinux −s /dev/nftla1**.

10.6. Step 5: reboot

Remove the floppy disk or unhook the floppy drive and reboot your Bering device. If the BIOS of the device is properly configured, Bering should now boot from the DoC.

10.7. Thanks to...

Jacques Nilo and Eric Wolzak for creating Bering, all the LEAF developers for their contributions, and Mike Noyes for his support of the LEAF project and great work to encourage continuous improvement.

Prev
Booting Bering from different boot−media

Home
Up

Next
Wireless and orinoco drivers

11. Wireless and orinoco drivers

11.1. Objectives

We want here so setup an internal wireless network that will share an internet access through a Bering firewall. We assume here that your external interface to the internet (eth0) is connected to your ISP via a standard NIC whereas your internal interface (eth1) to your network is connected through a wireless NIC.

What follows has not been tested by the author which does not have the corresponding hardware. Bob Pocius did the testing using an Orinoco Gold PCMCIA card connected to a PC through an ISA/PCMCIA adpater. Thanks to Bob for his help!

The most complete information on wireless under Linux can be found on Jean Tourrilhes web site. Jean is the developper of the wireless tools. He has also a very detailed page on Linux Orinoco drivers.

Comments on this section should be addressed to its maintainer: Jacques Nilo <jnilo@users.sourceforge.net>.

11.2. Step 1: declare the appropriate packages

First of all download the pcmcia_orinoco.lrp package from the Bering packages area and rename it pcmcia.lrp. This package is derived from the standard Bering pcmcia.lrp package and includes the orinoco drivers.

You then need to download the wireless.lrp and the the libm.lrp packages.

Depending on your ISP connection and your network hardware, declare the appropriate packages. For example:

- ppp, pppoe, and pcmcia if you connect through an ADSL/PPPoE connection and have a wireless NIC connected through a PCMCIA adapter.
- pcmcia if you connect through a fixed IP cable−modem ISP and have a wireless NIC connected through a PCMCIA adapter.
- pump, pcmcia if you connect through a dynamic IP cable−modem ISP and have a wireless NIC connected through a PCMCIA adapter.
- none of the above if you connect through a fixed IP cable−modem ISP and have a PCI native wireless card.

In the first case your `syslinux.cfg` file will look like (adapt to your own case):

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,ppp,pppoe,pcmcia,wireless,libm,keyboar
```

The last two lines ("default linux ... dnscache,weblet") must be typed as a single one in `syslinux.cfg`

11.3. Step 2: declare the appropriate modules

Declare the modules needed by the packages you are using: network modules and − if necessary − ppp modules.

The network and ppp modules are declared through the modules package configuration menu. Refer to the Bering installation guide.

The pcmcia modules are stored in the `/lib/modules/pcmcia` directory of the pcmcia package and loaded by the cardmgr program. Refer to the pcmcia section of the Bering user's guide.

Do not declare the pcmcia modules in the `/etc/modules` file. They will be automatically loaded from the `/lib/modules/pcmcia` directory by the `/etc/init.d/pcmcia` script.

Then backup the modules.lrp package and − if necessary − the pcmcia.lrp package.

11.4. Step 3: configure ppp

If your connection with your ISP needs PPP please refer to the Serial Modem section of this user's guide to learn how to configure your ppp package.

11.5. Step 4: configure pcmcia and wireless

The following modules are provided with the pcmcia_orinoco.lrp package:

```
# ls -la /lib/modules/pcmcia
drwxr-xr-x    2 root     root         4096 avr 25 08:22 ./
drwxrwxrwt   27 root     root         4096 avr 25 07:52 ../
-rw-r--r--    1 root     root        11248 avr 25 08:21 ds.o
-rw-r--r--    1 root     root         6060 avr 25 08:21 hermes.o
-rw-r--r--    1 root     root        33728 avr 25 08:21 i82365.o
-rw-r--r--    1 root     root         8100 avr 25 08:21 orinoco_cs.o
-rw-r--r--    1 root     root        42152 avr 25 08:21 orinoco.o
-rw-r--r--    1 root     root        57272 avr 25 08:21 pcmcia_core.o
```

Check that the PCMCIA modules provided in the pcmcia_orinoco.lrp package fit your needs. If not, download the appropriate modules from the Bering PCMCIA modules download area in the `/lib/modules/pcmcia` directory. Refer to the Bering installation guide to learn how to do that.

Enter the LEAF Package configuration menu and choose pcmcia. The following menu will appear

```
                      pcmcia configuration files

      1) pcmcia default parameters
      2) pcmcia configuration
      3) wireless configuration

 q) quit
 ------------------------------------------------------------------------------
      Selection:
```

Entry 1) allows to edit the `/etc/default/pcmcia` file which defines the pcmcia parameters that will be used by the cardmgr program and the `/etc/init.d/pcmcia` script.

In our practical example (an Orinoco gold card) this file will contain:

```
PCMCIA=yes
PCIC=i82365.o
PCIC_OPTS=
CORE_OPTS=
CARDMGR_OPTS=
```

You may need to specify something like: PCIC_OPTS=i365_base=0x3e2 if you are using an ISA/PCMCIA adapter.

Entry 2) allows to edit the `/etc/pcmcia/config.opts` file. The default file provided in the pcmcia.lrp package is the one provided in the pcmcia−cs package. It looks like:

```
include port 0xa00−0xaff
#
# Resources we should not use, even if they appear to be available
#
# First built−in serial port
exclude irq 4
# Second built−in serial port
exclude irq 12
# First built−in parallel port
exclude irq 7
```

Refer to the PCMCIA How−to for the explanation of the different options. In most cases you won't need to edit this file.

Entry 3) allows to edit the `/etc/pcmcia/wireless.opts` file which contains some templates for the most common drivers. Just fill in your card configuration in the template corresponding to your driver configuration. Then, to activate it, you need to remove or comment the four lines a the top of `wireless.opts`.

For an orinoco Gold card, this file will look like:

```
# Config info for Orinoco Wireless Cards
*,*,*,00:02:2D:*)
       INFO="Orinoco"
       MODE="Ad−Hoc"
       CHANNEL="1"
       RATE="11M"
       ESSID="Home"
       ;;
```

More information on the structure of the `wireless.opts` can be found here.

11.6. Step 5: configure your interfaces file

Trough the LEAF configuration menu type 1 to access to the network configuration menu and 1 again to edit your `/etc/network/interfaces` file. Enter the following information:

```
auto lo eth0

iface lo inet loopback

iface eth0 inet dhcp

iface eth1 inet static
        address 192.168.1.254
        masklen 24
        broadcast 192.168.1.255
```

We assume here that you get a dynamic IP from your ISP through pump. The corresponding interface (eth0) is brought up automatically at boot time (eth0 is in the auto statement). The wireless NIC is connected to eth1 and is assigned the 192.168.1.254 local address. This interface is NOT brought up automatically at boot time. The pcmcia package will start cardmgr through the `/etc/init.d/pcmcia` script executed at boot time. The cardmgr program will then call the `/etc/pcmcia/network` script which will in turn:

- Execute the `/etc/pcmcia/wireless` script after having read parameters from `/etc/pcmcia/wireless.opts`. This step will take care of iwconfig initialization before eth1 is up.
- Bring up the etc1 interface reading the info from the `/etc/network/interfaces` file

Backup the etc.lrp package.

11.7. Step 6: configure Shorewall

Check the Shorewall configuration as explained in the installation guide. The Bering default setup should be OK fo the above example.

11.8. Tips and tricks

ISA/PCMCIA adapters appear more stable for wireless cards than PCI/PCMCIA adapters. That is a good news for LEAF users who tend to use an old machine to setup their router.

If you do not succeed to activate your PCMCIA card while using it through a PCI/PCMCIA adpater you might give a try to the i82365.o patched module.

The wavelan2_cs Lucent driver for the orinoco card is also available in the Bering modules/pcmcia download section and can be used instead of the MPL/GPL orinoco_cs driver.

12. IPSec configuration

12.1. Objectives

This document assumes that you have a Bering Firewall with an internal interface on eth1 and an external interface on eth0, and that you want to accept IPSec connections from Windows 2000 machines ("roadwarrior" clients or gateways for subnets) on the external interface, then treat those external clients or subnets as members of your internal network.

Also, there is a sizeable portion of this document that covers the configuration of the Windows 2000 IP Security Policy Utility. Please do not let this part slow you down if you are not interested in interoperating with Windows 2000 clients. It is extremely long, and I only wrote it down because most of what I found on the internet about it was pure "click here − click there" stuff and didn't really explain what was going on or the ramifications of "clicking there." I spent a lot of time trying to figure out the dark mysteries of their user interface, so hopefully, no one else will have to wear out their mouse finger trying to do so.

There are more complex configurations than this, which you should be able to understand better after reading.

Comments on ths section should be addressed to its maintainer: Chad Carr <ccarr@franzdoodle.com>.

12.2. Step 1: load ipsec or ipsec509 package

Copy the ipsec.lrp or ipsec509.lrp package to the floppy. Also, you must copy the mawk.lrp package since it is needed by the ipsec scripts. You do not need the ifconfig.lrp package

You may need some space to store the packages and the ipsec module and generally a single floppy won't be enough. Check the Bering user's guide section about "Booting Bering from different boot−media" for tips.

Modify the syslinux.cfg file to load the new packages. It might look like this:

```
display syslinux.dpy
timeout 0
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos
        PKGPATH=/dev/fd0u1680 LRP=root,etc,local,modules,pump,keyboard,shorwall,dnscache,weblet
```

The last two lines ("default linux ... ipsec,mawk") must be typed as a single one in `syslinux.cfg`

Copy the ipsec.o module from the modules package which matches your kernel. Don't even try to do this with mismatching modules, kernel or ipsec utilities!

Install this module using the method described in the main Bering documentation.

12.3. Step 2: generate certificates with openssl

Certificates usually need to be generated from the host machine since the router usually doesn't have enough randomness to generate them easily. I use the Debian package, and I assume there is a RedHat package. Here

is a link to a document decribing how to compile it from source.

If you do not desire to use certificates (you only wish to use preshared keys) you may skip to Step 4

Make a new certificate authority

```
# mkdir -p demoCA/private; mkdir -p demoCA/newcerts;
# touch demoCA/index.txt; echo 01 >> demoCA/serial; chmod -R 700 demoCA
# openssl req -x509 -days 3650 -newkey rsa:2048 -keyout demoCA/private/cakey.pem -out demoCA/ca
# openssl ca -gencrl -out crl.pem
```

Make your ipsec server certificate

```
# openssl req -newkey rsa:2048 -keyout serverKey.pem -out serverReq.pem
# openssl ca -policy policy_anything -in serverReq.pem -days 1825 -out serverCert.pem -notext
# openssl x509 -in serverCert.pem -outform DER -out x509cert.der
# fswcert -k serverKey.pem > ipsec.secrets
```

Make your client certificates

```
# openssl req -newkey rsa:2048 -keyout clientKey.pem -out clientReq.pem
# openssl ca -policy policy_anything -in clientReq.pem -days 1825 -out clientCert.pem -notext
# openssl pkcs12 -export -inkey clientKey.pem -in clientCert.pem -certfile demoCA/cacert.pem -c
```

Put all of this onto your Bering floppy or compact flash card, unmount it and boot it

12.4. Step 3: boot Bering and move certificates into place

Put cacert.pem onto your Bering box in the /etc/ipsec.d/cacerts directory (you will have to create this with mkdir). Put crl.pem into the /etc/ipsec.d/crls directory (make this one, too). Put x509cert.der into /etc. Get the info in ipsec.secrets into your /etc/ipsec.secrets file like so:

```
# echo ipsec.secrets >> /etc/ipsec.secrets
```

12.5. Step 4: configure ipsec.conf

An ipsec.conf file, you'll find, is a very personal thing. A very vanilla setup using preshared keys would look like the following:

```
config setup
        interfaces=%defaultroute
        klipsdebug=none
        plutodebug=none
        plutoload=%search
        plutostart=%search
        uniqueids=yes

conn %default
        keyingtries=0
        authby=secret
        left=<router ip address>
        leftsubnet=<internal subnet>
        leftfirewall=yes
```

```
        pfs=yes
        auto=add

conn w2k-road-warriors
        right=%any
```

There is really no substitute for reading the man page, however.

With certificates, the same setup would look like this:

```
config setup
        interfaces=%defaultroute
        klipsdebug=none
        plutodebug=none
        plutoload=%search
        plutostart=%search
        uniqueids=yes

conn %default
        keyingtries=0
        authby=rsasig
        leftrsasigkey=%cert
        rightrsasigkey=%cert
        left=<router ip address>
        leftsubnet=<internal subnet>
        leftid="C=US, ST=CA, O=ipsecgw, CN=me, Email=you@yourdomain"
        pfs=yes
        auto=add

conn w2k-road-warriors
        right=%any
```

A couple of things to watch out for. 1) Do not put apostrophes or single quotes in any of your distinguished name fields! It causes blindness and other very bad things to happen. 2) Make sure that the date on your router is between the notBefore and notAfter dates on all your certificates!

12.6. Step 5: configure ipsec.secrets

If you wish to use preshared keys, your ipsec.secrets should look like the following:

```
%any <router ip address>: PSK "<your preshared key>"
```

if you are dealing with roadwarriors with dynamic ip addresses. If you know the ip address of the endpoint and you do not wish to share the same key amongst multiple roadwarriors, you have the option of specifying the ip address instead of "%any." If you don't want to share keys, and you don't kow the ip addresses of your clients, certificates are your only real option.

Your ipsec gateway's certificate can either have its private key extracted (using fswcert as in Step 2) and put in the ipsec.secrets file, or it can be stored in the /etc/ipsec.d/private directory (in either der or pem format) and be referenced in ipsec.secrets by filename with an optional passphrase.

If you choose to extract the key and keep the whole thing in ipsec.secrets directly, your ipsec.secrets file will look like this:

```
: RSA {
        Modulus:         0xB664D963F28A...
        PublicExponent:  0x010001
        PrivateExponent: 0x518CA9BE0C55...
        Prime1:          0xED48CBD214FC...
        Prime2:          0xC4C7B7244774...
        Exponent1:       0x314D4BD435BA...
        Exponent2:       0x6237A8E2B3C3...
        Coefficient:     0xCEA15F52310E...
  }
```

Except the long strings of gibberish will be much longer. The `: RSA` must start at the left margin, but every other line must be indented (spaces or tabs will do). The file MUST have no more than 700 permissions and be owned by root to be secure.

Otherwise, put the private key (serverKey.pem from Step 2) in /etc/ipsec.d/private, secure it with an optional passphrase (recommended) and reference it in the ipsec.secrets file like so:

```
: RSA serverKey.pem "<optional passphrase>"
```

12.7. Step 6: configure Shorewall

You need to add a new zone to shorewall to handle hosts that connect through ipsec, and also add a tunnel definition to allow the udp port 500 traffic for Internet Key Exchange (IKE) and protocols 50 and 51 (ESP and AH) that are used for the IPSec payloads.

You must not turn on route filtering for any interfaces involved in ipsec. The "Bering recommended" way to turn this off is to use the `/etc/network/options` file and change the "spoofprotect" parameter to "no"

Add the gw zone to the `/etc/shorewall/zones` file

```
gw      ipsec0
```

Then use an entry like this in the `/etc/shorewall/tunnels` file

```
ipsec   net     0.0.0.0/0       gw
```

Use the ip address of the ipsec endpoint if you have it, because that will be more secure.

12.8. Step 7: configure Windows 2000 client

Configuring Windows to do this same thing is much harder. I would say that until you have done it properly once, it borders on black magic. Even if you have done it properly once, if the configuration is even slightly different and you didn't take the time to really understand it the first time, you are in for another rough ride. The way your mouse finger feels after clicking your way through the dialogs for this configuration is just another symbol of how most complicated things are easier and more user friendly in Linux.

It helps to have a custom "management console" when you're dealing with ipsec. You can put this on the desktop or someplace else convenient, and save your mouse finger from exhaustion clicking through menus to find things. Use the following steps:

Start − Run − mmc Console − Add/Remove Snap−in Add − Certificates Add − Computer Account − Local Computer − Finish Add − IP Security Policy Management Add − Local Computer − Finish − Close − OK Console − Save − <wherever you want to put it>

You can just double click on the icon this creates to open the custom console from now on.

In order to configure Windows 2000, there are several basic entities that you must understand. It is easy to get lost in all the clicky−clicky:

```
        IP Security Rules                − the highest level of granularity.
                                           IP Security Rules are composed of:
                an IP Filter List        − which packets match the rule?  An
                                           IP Filter list is composed of:
                        Filters          − traditional ip address, subnet mask,
                                           protocol or port filtering, like
                                           ipchains
                a Filter Action          − what do we want to do with those
                                           packets? Encrypt? Sign?  A Filter
                                           Action is composed of:
                        Security Methods − Different negotiable combinations of
                                           signing and encrypting.  FreeS/WAN
                                           works in ESP mode with 3DES
                                           encryption and MD5 signing.  This is
                                           a custom setting in Windows.
                Authentication Methods   − how do we authenticate the players?
                                           Windows can do Kerberos, x.509
                                           certificates from a CA (that can be
                                           you!) or preshared keys
                a Tunnel Setting         − is this a tunnel?  what is the
                                           endpoint IP Address?
                a Connection Type        − does this IP Security Rule apply to
                                           all network connections, or just lan
                                           or dialup connections?
```

Also, for Windows 2000, you must have the Service Pack 2. It will not do the required 3DES encryption without it. You can get it from:

http://www.microsoft.com/windows2000/downloads/servicepacks/sp2/sp2lang.asp

Pretty hairy. For what I wanted to do (tunnel mode ESP with MD5 signing and preshared key authentication) I had to set up two rules, one for inbound traffic specifying the Windows client IP address as the endpoint of the tunnel, and one for outbound traffic specifying the router as the endpoint of the tunnel. I did not want to have to know the IP address of the client, since I want to use DHCP to deliver these addresses, but I haven't worked a way around it yet. Maybe if some Windows people are reading this they can drop a line...

```
Configure the Windows 2000 client

        a) run the custom mmc console you just made
        b) click on ipsec security policies in left pane
        c) action − create IP security policy
        d) next, choose name (Win2k to FreeS/WAN), uncheck default response
           rule, check edit properties, finish
        e) add IP security rule to grab outbound traffic and tunnel it to
           FreeS/WAN using 3DES and MD5
        f) next, enter tunnel endpoint (<router IP Address>), lan connection,
           preshared key(<your preshared key>)
```

```
g) add both ip filter lists for inbound and outbound traffic, then you
   can just click on inbound traffic when you're defining that security
   rule
h) add ip filter list, name: "outbound traffic", add filter
i) next, src: my ip address, dest: any ip address, any proto, finish
        note: My setup is made to tunnel ALL ip traffic through my
        router.  If you are just tunneling traffic to one subnet, you
        should specify that here with the network address and subnet
        mask
j) add another filter list, name "inbound traffic", add filter
k) next, src: any ip address, dest: my ip address, any proto, finish, close
        note: see note above
l) select the "outbound traffic" filter list, next
m) add filter action to encrypt and authenticate with freeswan (3DES and MD5)
n) next, name "freeswan compatible", negotiate, do not communicate
   non-ipsec, custom (ESP, MD5, 3DES), edit properties, finish
o) uncheck allow unsecured but always respond..., check perfect forward
   security, OK
p) select the "freeswan compatible" filter action
q) uncheck edit properties, finish

The next one is easier, because you have already defined the filter
lists and filter action during the previous wizards, so you can just
select them to apply them to the inbound traffic IP Security Rule.
I'll go through this quickly, but just hit next or okay if I've skipped
a step.

r) add another IP security rule
s) next, tunnel endpoint (<client IP Address>) lan connection,
   preshared key (<your preshared key>), inbound traffic, freeswan
   compatible, finish
t) general tab, advanced, check master key perfect forward security,
   close, close
```

You should be done. Right click the "Win2k to FreeS/WAN" IP Security Policy and click assign in the context menu. Bring up a DOS window. Ping your router. If everything is correct, you will get "Negotiating IP Security." as the response to the first four pings, then should be pinging clear after that.

If not, double click on the "Win2k to FreeS/WAN" IP Security Policy to reenter configuration dialogs. You will see the two IP Security Rules you just created. Double click on one of them to check the configuration.

You will see five tabs at the top of the dialog corresponding to the items described at the beginning of this section. First check the "outbound traffic" filter list, then the "inbound traffic" filter list. Double click on them to enter the configuration dialog, then double click the Filter. Do they have the right source and destination addresses? Remember, this will match packets similar to the way ipchains rules do, so if the rule doesn't match properly, then you will not "forward" the packet to the Filter Action, and it will not get encrypted properly.

If the Filter Lists are both okay, move on to the Filter Action. We have the same Filter Action for both IP Security Rules, so we just have to make sure that it says to negotiate security with ESP, 3DES and MD5. We should also ensure again that "Accept unsecured communication..." and "Allow unsecured communication..." are unchecked, 'cause those are not going to accomplish what we are trying to get, and that "Session key Perfect Forward Secrecy" is checked.

Check the Authentication Method to make sure that the shared key is exactly the same as the ipsec.conf file and that there is not a carriage return at the end.

Check the tunnel setting and make sure that the tunnel endpoint is the router ip address for the "outbound traffic" Filter List and the ip address of the Windows 2000 Client for the "inbound traffic" list.

The Connection Type should be LAN only. We don't want to inadvertently try to encrypt our dialup sessions, do we?

If any of these things was wrong, you will have to restart the IPSEC Policy Agent service by clicking Start − Control Panel − Services, right clicking on the service and clicking Restart.

Prev | Home | Next
Wireless and orinoco drivers | Up | Monitoring Bering through a terminal console

13. Monitoring Bering through a terminal console

13.1. Objectives

We assume here that you want to monitor Bering through − say − a minicom terminal attached to the first serial port of your router (com1/ttyS0). That is a frequent situation with LEAF routers which, very often, do not have a screen attached to them.

Comments on this section should be addressed to its maintainer: Jacques Nilo <jnilo@users.sourceforge.net>.

13.2. Step 1: Modify `/etc/inittab` and `/etc/securetty` files

Through the LEAF configuration menu type 2) to get access to the "System configuration" menu:

```
                        System configuration menu

       1) Master LRP settings. (lrp.conf)
       2) POSIXness settings (POSIXness.conf)
       3) File system mounts. (fstab)
       4) Lowest level boot-up configuration (inittab)
       5) System wide profile (profile)
       6) Ports root is allowed to login to. (securetty)
       7) System logging configuration. (syslog.conf)
       8) Service name to number translation (services)
       9) Local timezone TZ setup (tzvalue)

  q) quit
  ------------------------------------------------------------------------------
        Selection:
```

Enter 4) to edit `inittab`. Comment out getty's on tty1 and tty2 and uncomment getty on ttyS0 (com1). For access through com2, com3 or com4 replace by ttyS1, ttyS2 and ttyS3 respectively.

Your `inittab` file will look like:

```
<snip>
# Format:
#  <id>:<runlevels>:<action>:<process>
#1:2345:respawn:/sbin/getty 38400 tty1
#2:23:respawn:/sbin/getty 38400 tty2
#3:23:respawn:/sbin/getty 38400 tty3
#4:23:respawn:/sbin/getty 38400 tty4
#5:23:respawn:/sbin/getty 38400 tty5
#6:23:respawn:/sbin/getty 38400 tty6

# Example how to put a getty on a serial line (for a terminal)
#
T1:23:respawn:/sbin/getty -L ttyS0 19200 vt100
<snip>
```

Enter 6) to edit `/etc/securetty` to add ttyS0. Your file will look like:

```
# /etc/securetty: list of terminals on which root is allowed to login.
# See securetty(5) and login(1).
#
# Include ttyp0, ttyp1, etc to allow telnet access. *NOT RECOMMENDED*
ttyS0
tty1
tty2
tty3
tty4
tty5
tty6
tty7
tty8
```

Once this is done, backup etc.lrp

13.3. Step 2: Modify your `syslinux.cfg` file

Edit the `syslinux.cfg` file on your floppy and add the two following statements:

- serial 0 19200 at the top of your file
- append console=ttyS0,19200

The syntax of the *serial* statement is as follows:

SERIAL port [baurate]. This enables a serial port to act as the console. "port" is a number (0 = ttyS0 = com1, etc.). If "baurate" is omitted, the baud rate defaults to 9600 bps. The serial parameters are hardcoded to be 8 bits, no parity, 1 stop bit.

The append statement add one or more options to the kernel command line.

Your `syslinux.cfg` file will look like:

```
serial 0 19200
display syslinux.dpy
timeout 0
append console=ttyS0,19200
default linux initrd=initrd.lrp init=/linuxrc root=/dev/ram0 boot=/dev/fd0u1680:msdos PKGPATH=/
```

13.4. Step 3: reboot...

Connect a cable to the serial port of your router and open a terminal on your monitoring machine. You should be then able to control your Bering router from that console.

One application you can use to connect to your router's serial port is minicom, but you'll need to change the default settings since you won't be talking to a modem. As root, launch 'minicom −s'. Change the speed (in serial port setup) to 19200. Then change the modem init string (in modem and dialing) to "~^M~". Save the settings as something other than df1 (I use "leaf"), quit, and relaunch (not as root) using 'minicom leaf'.

14. Time in Bering

14.1. Objectives

These instructions are for those who want to setup properly the system time of their Bering box either using the old "rdate" function (available on the Bering floppy) or using the more precise, up−to−date ntpdate client (available as a separate package). You will also learn how to transform your Bering box in a time server in order to synchronize time of your internal network.

Many thanks to Jeff Newmiller from whom we stole a significant part of a mail contribution to the leaf−user list and for the time he spent improving and proofreading the initial version of this chapter.

Comments on this section should be addressed to the maintainers: Jacques Nilo <jnilo@users.sourceforge.net> or Eric Wolzak <leaf@wolzak.de>.

14.2. Define your timezone

a/ Obtain the appropriate zoneinfo file for your timezone.

This binary file will contain generalized rules for converting between GMT and your local time. One location where these files are kept is here, but their format has not changed in a long time and is not expected to change anytime soon so you can pull one from the Linux distribution of your choice.

b) Copy this file to the Bering ramdisk as `/etc/localtime`. On most conventional Linux distributions, `/etc/localtime` would be a symbolic link to the appropriate file in `/usr/share/zoneinfo/`, but that directory is not contained in etc.lrp and having symbolic links across packages is not recommended.

c) Use the "date" command to confirm that the zoneinfo file is behaving as desired.

14.3. Set the system date/time

There are three common methods to do that:

- Method 1: Reboot the machine and set the time in the BIOS. Note that for a pure linux machine like a router it only makes sense to set the bios clock to UTC (GMT0).
- Method 2: Set the Linux time with the "date [MMDDhhmm[[CC]YY][.ss]]" command and back it up to the CMOS clock with "hwclock". For example, if it is 9:05:15 pm on Jan 31, 2002, then you would use:

```
# date 013121052002.15
# hwclock −−systohc
```

  hwclock will set the CMOS clock to UTC.
- Method 3: Set the Linux time with "rdate {timeserver}" and back it up to the CMOS clock with "hwclock". For example, if you have a Linux box at 192.168.1.3 (see step f), you can use:

```
# rdate -s 192.168.1.3
# hwclock --systohc
```

Note that if you don't have a Linux workstation available, you can use a program like AboutTime for Windows to set the Windows machine time correctly, and while AboutTime is running and its server options are enabled you can use "rdate" against that machine.

For the rdate command to work you will have to open the "time" service (tcp/37) from your firewall to the internet. Edit the Shorewall rules file and add:

```
ACCEPT          fw      net     tcp     time
```

14.4. Edit the contents of `/etc/timezone` (optional)

This will describe your timezone. I am not aware of any packages used with Bering that depend on this file, but it might as well be consistent.

14.5. Activate daily clock updating (optional)

Here you have to choose one of the three following options (mutually exclusive):

- You can activate daily clock updating via rdate (tcp port 37) to a nearby "time" protocol server. This service is handled internally by the inetd daemon on a *nix workstation. To activate rdate updating, edit `/etc/lrp.conf` to specify the ip number of the desired time server for lrp_DATE_SERVER, and uncomment this variable (See the "configure your system" chapter of the Bering installation guide). The main advantage of this option is that you do not need an extra package to synchronise time on your Bering box. The main drawbacks are a/ rdate is not accepted by every public time servers and b/ rdate is less precise than NTP (see below).
- Download the ntpdate.lrp package from the Bering package download area and add it to your syslinux.cfg file. It will provide you with the NTP client from http://www.ntp.org. Then go to the ntpdate configuration menu and declare the timeservers you want to query and, optionnally the frequency of the update in the ntpdate cronjob parameters (default= every hour)
- Activate your own time server. In this scenario you will have to download the ntpsimpl.lrp and the libm.lrp packages from the Bering packages download area and add them to your syslinux.cfg file. You will have the ntpd daemon from http://www.ntp.org (ntpd is a more recent version of xntpd). Then go to the ntpsimpl configuration menu and declare the timeservers you want to query.

Do not forget to adjust your firewall to allow access to NTP services:

```
ACCEPT          fw      net     udp     ntp    (if you want to query an external NTP server
ACCEPT          loc     fw      udp     ntp    (if you want to query your Bering box time s
```

A list of available public timeservers is available here. All of them will accept request from ntpd or ntpdate. Only a few of them will accept rdate requests.

14.6. Internal network NTP clients

If you have a time server running on your Bering box, you might be looking for NTP clients for your internal network machines.

For your internal network linux boxes, ntpdate will do.

If you are running Windows machine of any variety, you can have a look at Tardis or Automachron.

14.7. Miscellaneous

- All lines in `/etc/tzvalue` can be commented out if you provide proper zoneinfo file. See section 2 of this chapter.
- Either reboot, or restart logging (svi sysklogd restart) to cause times recorded by the system logger to use the new timezone information.

Prev
Monitoring Bering through a terminal console

Home
Up

Next
The Bering "mail" and "cron" facilities

15. The Bering "mail" and "cron" facilities

15.1. Objectives

This section should be an help to use two special features of your LEAF Bering box namely the mail and cron facilities.

This document is maintained by Eric Wolzak <leaf@wolzak.de>.

15.2. The mail command

In the Bering root.lrp there is a mail command, which is one of the "POSIXness" script files. With this command you can send emails typed directly from the console or written as files. You can also send file attachment.

This command can only be used to send mail and differs substantially from the "real" linux mail command. In particular you cannot edit (read, delete, ..) mail from the firewall. As an alternative to this simple mail program you can use a real mailer program like qmail

The syntax of the Bering mail command is:

```
# mail
Usage: mail options to[,...]
Options: [-s subject] [-c cc[,...]] [-b bcc[,...]]
         [-a attachment[,...]] [-d domain] [-h smptserver]
```

- -a attach text file(s)
- -d specify from FQDN, overriding local domain
- -h specify SMTP server, overriding the MAIL_SERVER setting
- -v verbose

Mail default settings are set in `/etc/POSIXness.conf`. Please refer to the Bering installation guide (System configuration section) for detailed instructions about default mail parameters

Through the System Configuration menu choose the (2) POSIXness Configuration entry. You will then be able to set the following options:

- MAIL_SERVER: this is the SMTP server where mail is sending its mail to (e.g.MAIL_SERVER="smtp.myprovider.com")
- MAIL_DOMAIN: this is the domain which will be shown in the from list (e.g. MAIL_DOMAIN="yourdomain.org" the FROM line will then be root@yourdomain.org)
- USER: this is the user you will use as the "part" before the @ sign for yourdomain.org. If you don't set a name here then the mail will be sent with the user the mail command is evoked or defaulting to root.If USER="john.doe", then your mail will be from: john.doe@yourdomain.org

Be careful about the MAIL_DOMAIN definition as lots of smtp servers will refuse mails with a name they cannot resolve to a valid IP. Others refuse to relay mails that cannot be delivered locally.

To mail a message to someone edit a file with the editor (e.g. ae message), type your text and save the message file. Then to send your message:

```
cat message | mail −s "I want to tell you" to yourfriend@hisdomain.com
```

or as an attachement:

```
mail −s "I want to send you" −a "message" to yourfriend@hisdomain.com
```

You can also type your mail directly from the console:

```
mail −s "I want to send you" to yourfriend@hisdomain.com
```

Once you hit return, the console will wait for a message to be typed in. Once you have finished with your message input, type CTRL−D.

To mail the log and alerts files to the Bering box admin, set lrp_MAIL_ADMIN to the email address you want your logfiles sent to you. This parameter is found in the Master LRP settings entry of the System configuration menu

To be able to send mail from the firewall, you will need to open port TCP/25 of the firewall. In the shorewall rules file you will need to include the following statement:

```
ACCEPT          fw      net     25      tcp
```

15.3. Cronjobs

The cronjobs are executed according to the entries defined in the directories `/etc/cron.d/` (every minute), `/etc/cron.daily` (every day), `/etc/cron.weekly` and `/etc/cron.monthly`.The most important part to add things to is probably `/etc/cron.d`. The syntax is the standard syntax as is read in man cron and crontab.

```
#Periodic schedule for multicron. (Ping check, Space check, etc)
#Default: Every 15 minutes
*/15    * * * * root    /etc/multicron−p
*       * * * * root    /bin/date >>/tmp/tijd
*/2     * * * * root    /bin/beep −f 1200
```

In this example multicron−p is executed every fifteen minutes, date every minute etc. After changing the cronjob is updated automatically. You can verify this with

```
# tail −f /var/log/syslog
Aug 18 09:15:01 firewall /USR/SBIN/CRON[28891]: (root) CMD (/etc/multicron−p)
Aug 18 09:16:01 firewall /USR/SBIN/CRON[9097]: (root) CMD (/bin/beep)
Aug 18 09:16:01 firewall /USR/SBIN/CRON[29944]: (root) CMD (/bin/date >>/tmp/tijd)
Aug 18 09:16:01 firewall /usr/sbin/cron[1774]: (*system*multicron) RELOAD(/etc/cron.d/multicron
```

Important is that you have one empty line after the last entry in the cron file. You can edit the multicron file as above or (probably a better idea) insert a new file with the syntax as before for each additional purpose.

An example for this could be:

```
#ls /etc/cron.d
-rw-r--r--    1 root     root          211 Aug 18 09:15 multicron
-rw-r--r--    1 root     root           93 Aug 18 09:13 closewindows
-rw-r--r--    1 root     root           80 Aug 17 08:12 md5sumfiles
```

Prev
Time in Bering
Home
Up